Run the code online here:

https://onlinegdb.com/N-6KusNbR

Code:

```
//Modified Euler's Method for differential equations
#include<iostream>
#include<math.h>
#include<iomanip>
using namespace std;
double df(double x, double y)
{
    double a=x+y;            //function for defining dy/dx
    return a;
}
int main()
{
    double x0,y0,x,y_i,dy1,dy2,dy_avg,y_n,h;    //for initial values, width, etc.
    cout<<"\nEnter the initial values of x and y respectively:\n"; cin>>x0>>y0;
//Initial values
    cout<<"\nEnter the value of x for which you would like to find y:\n"; cin>>x;
    cout<<"\nEnter the interval width,h:\n"; cin>>h;                        //input
width
cout<<"x"<<setw(16)<<"y"<<setw(16)<<"hy'"<<setw(16)<<"y_new"<<setw(16)<<"hy_new'"<<setw
(16)<<"hy'avg"<<setw(16)<<"y_n+1"<<endl;
    cout<<"-------------------------------------------------------------------------------
----------------------\n";
    while(fabs(x-x0)>0.0000001)      //I couldn't just write "while(x0<x)" as they both
are floating point nos. It is dangerous to compare two floating point nos. as they are
not the same in binary as they are in decimal. For instance, a computer cannot exactly
represent 0.1 or 0.7 in binary just like decimal can't represent 1/3 exactly without
recurring digits.
    {
        dy1=h*df(x0,y0);        //calculate slope or dy/dx at x0,y0
        y_i=y0+dy1;            //calculate new y, which is y0+h*dy/dx
        dy2=h*df(x0+h,y_i);        //calculate slope or dy/dx at x0+h (new x),new y
        dy_avg=(dy1+dy2)/2.0;        //calculate the average of the slopes at y0 and
new y
        y_n=y0+dy_avg;            //calculate new y, which is y0+h*average(dy/dx)
cout<<x0<<setw(16)<<y0<<setw(16)<<dy1<<setw(16)<<y_i<<setw(16)<<dy2<<setw(16)<<dy_avg<<
setw(16)<<y_n<<endl;
        x0=x0+h;            //calculate new x.
        y0=y_n;                //pass this new y as y0 in the next iteration.
    }
    cout<<x0<<setw(16)<<y0<<endl;
    cout<<"The approximate value of y at x=0 is "<<y0<<endl;    //print the solution.
    return 0;
}
```

Run the code online here:

https://onlinegdb.com/N-6KusNbR

Sample output

```
Enter the initial values of x and y respectively:
0       1

Enter the value of x for which you would like to find y:
1

Enter the interval width,h:
0.1
x               y             hy'           y_new         hy_new'          hy'avg           y_n+1
-------------------------------------------------------------------------------------------------
0               1             0.1             1.1            0.12            0.11            1.11
0.1            1.11           0.121           1.231          0.1431         0.13205          1.24205
0.2         1.24205        0.144205        1.38626        0.168626        0.156415         1.39847
0.3         1.39847        0.169847        1.56831        0.196831        0.183339          1.5818
0.4          1.5818         0.19818        1.77998        0.227998        0.213089         1.79489
0.5         1.79489        0.229489        2.02438        0.262438        0.245964         2.04086
0.6         2.04086        0.264086        2.30494        0.300494         0.28229         2.32315
0.7         2.32315        0.302315        2.62546        0.342546         0.32243         2.64558
0.8         2.64558        0.344558        2.99014        0.389014        0.366786         3.01236
0.9         3.01236        0.391236         3.4036         0.44036        0.415798         3.42816
1         3.42816
The approximate value of y at x=0 is 3.42816
```

Exact (ideal) solution

dy/dx=x+y , y(0)=1

NATURAL LANGUAGE | MATH INPUT | EXTENDED K

Input interpretation

$$\left\{\frac{\partial y(x)}{\partial x} = x + y(x),\ y(0) = 1\right\}$$

d'Alembert's equation

$$y(x) = x\,(-1) + y'(x)$$

ODE classification

first-order linear ordinary differential equation

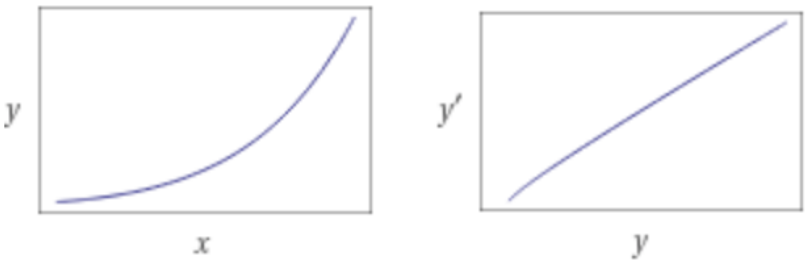
Alternate form

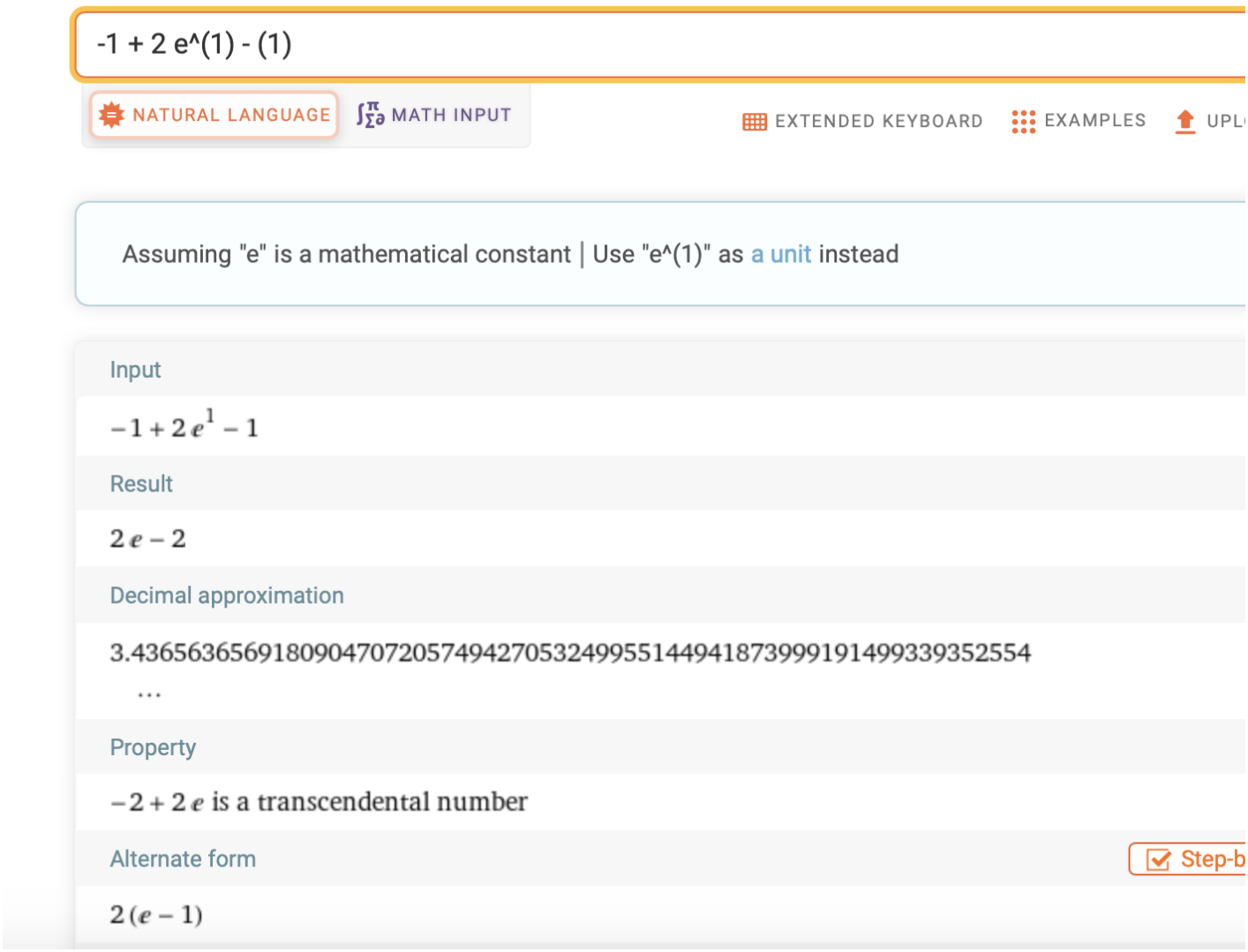
$$\{y(x) + x = y'(x),\ y(0) = 1\}$$

Differential equation solution

$$y(x) = -x + 2\,e^{x} - 1$$

Plots of the solution



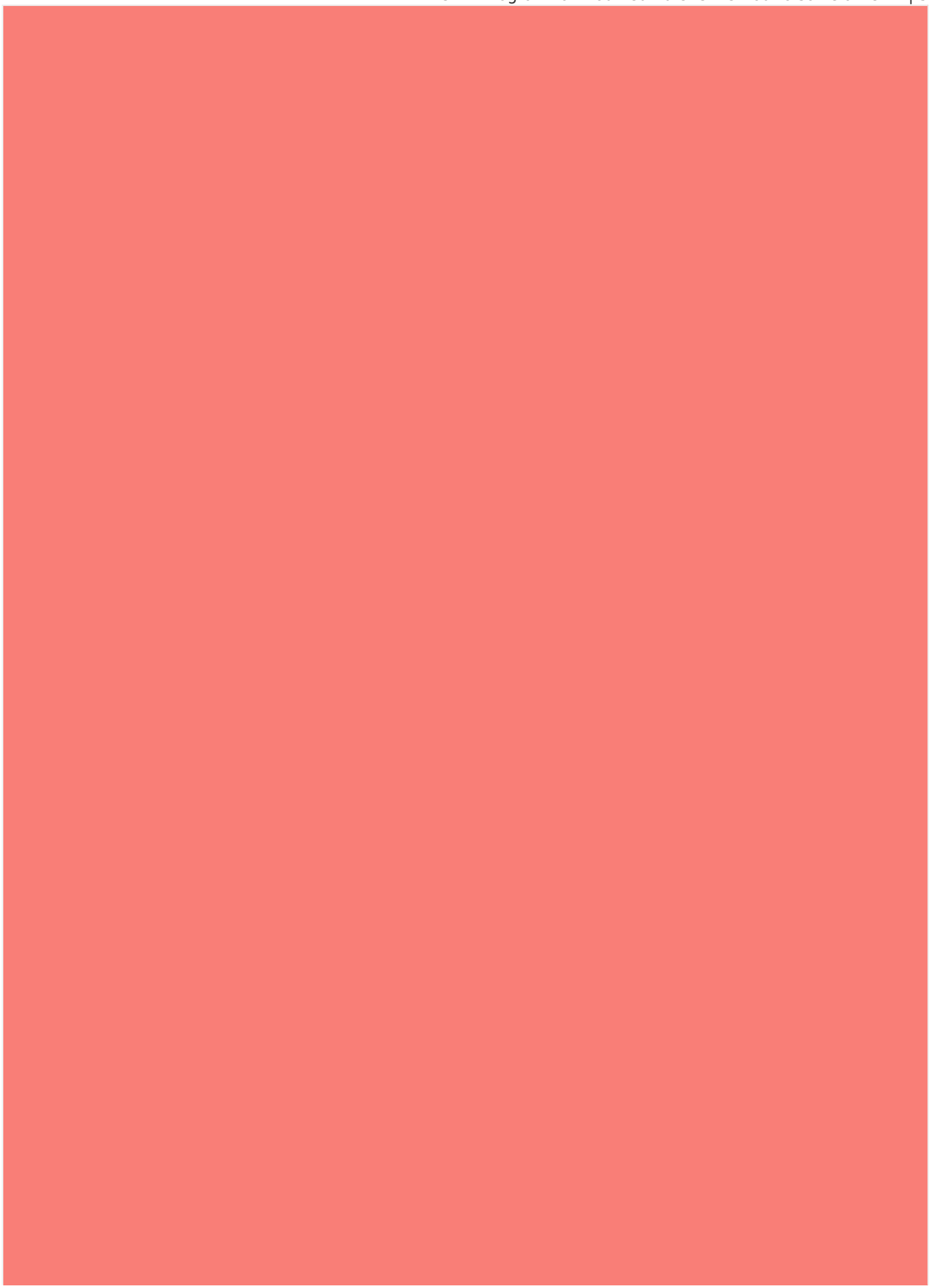


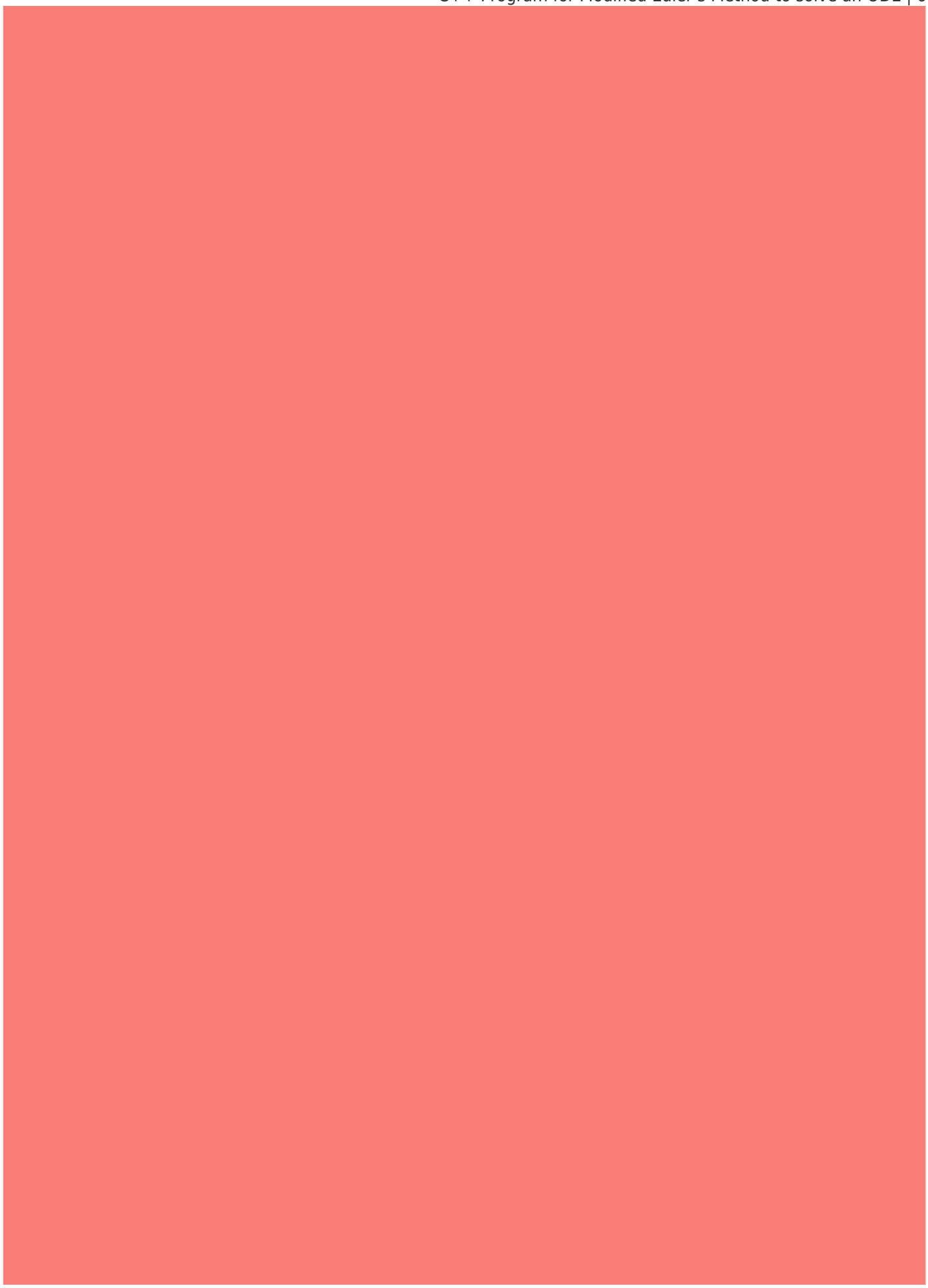
Explanation of the code:

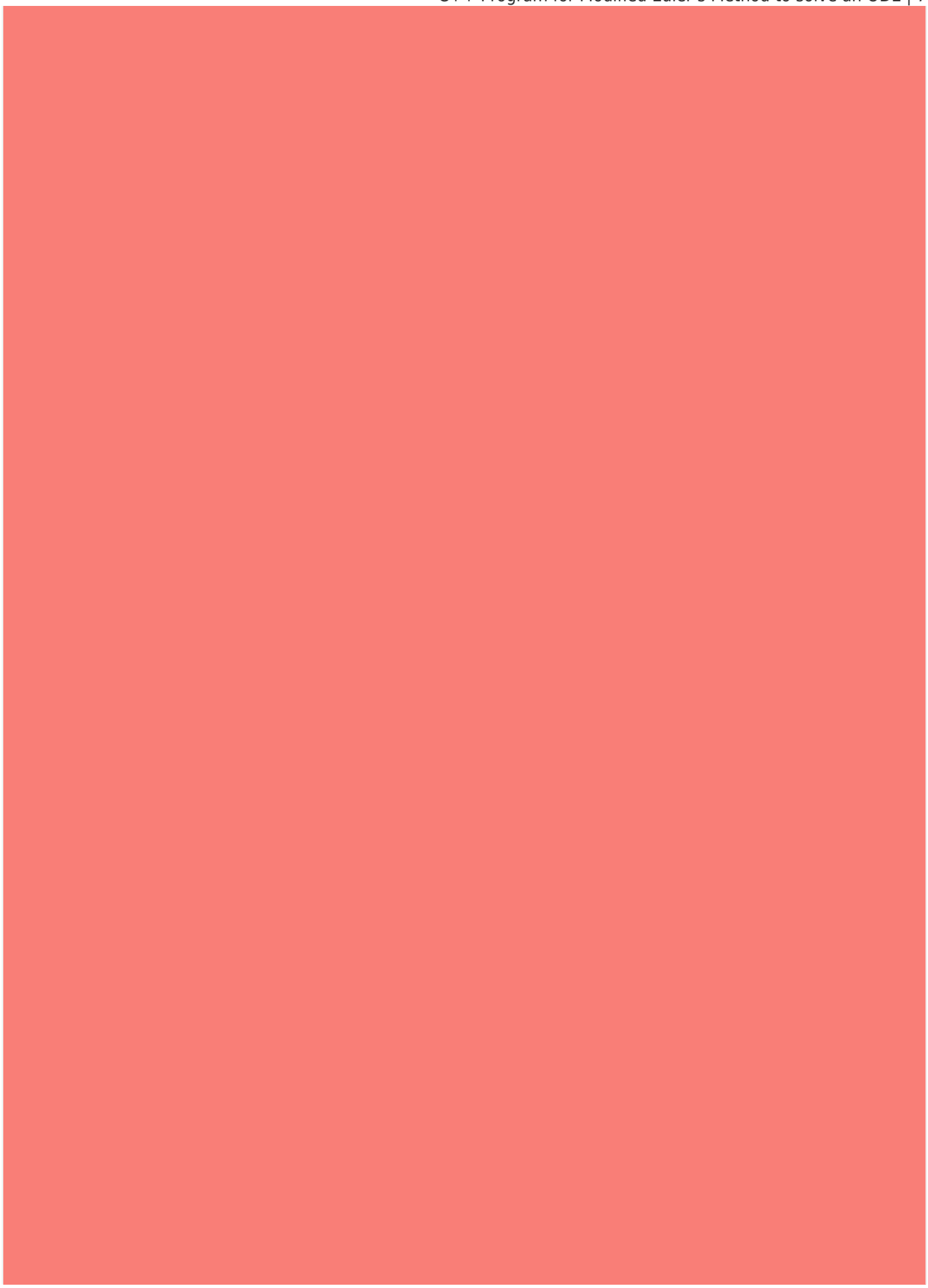
Manas Sharma

I'm a physicist specializing in computational material science with a PhD in Physics from Friedrich-Schiller University Jena, Germany. I write efficient codes for simulating light-matter interactions at atomic scales. I like to develop Physics, DFT, and Machine Learning related apps and software from time to time. Can code in most of the popular languages. I like to share my knowledge in Physics and applications using this Blog and a YouTube channel.

manas.bragitoff.com/







Share this:

Click to share on Facebook (Opens in new window)

Click to share on Twitter (Opens in new window)

Click to share on WhatsApp (Opens in new window)

Click to share on Pinterest (Opens in new window)

Click to share on Reddit (Opens in new window)

Click to share on LinkedIn (Opens in new window)

Click to email a link to a friend (Opens in new window)

Click to print (Opens in new window)

Click to share on Tumblr (Opens in new window)

Click to share on Pocket (Opens in new window)

Click to share on Telegram (Opens in new window)

[wpedon id="7041" align="center"]