

METODY NUMERYCZNE

LISTA 7

WIOLETTA ŁUPKOWSKA, 244831

CZWARTEK, 9:15

1.1)

```
double xEulerPrevious=1;  
double xEulerNext=0;
```

Klasa umożliwiająca całkowanie równań różniczkowych zwyczajnych pierwszego rzędu metodą Eulera:

```
import java.util.ArrayList;  
  
public class Euler {  
    Licz licz= new Licz();  
  
    public ArrayList liczEuler(double przedzialL, double przedzialU, double krok){  
        ArrayList<Double> listXEuler = new ArrayList();  
        ArrayList<Double> listXTrue = new ArrayList();  
        ArrayList<Double> listT = new ArrayList();  
        double t=przedzialL;  
        double xTrue;  
        double xEulerPrevious=1;  
        double xEulerNext=0;  
        do{  
            listT.add(t);  
            xTrue=licz.xTrue(t);  
            listXTrue.add(xTrue);  
            xEulerNext= Math.abs(xEulerPrevious + licz.ft(xEulerNext,t)*krok);  
            listXEuler.add(xEulerNext);  
            xEulerPrevious=xEulerNext;  
            t+=krok;  
        }while(t<przedzialU);  
        System.out.println("ListXTrue: ");  
        System.out.println(listXTrue);  
        return listXEuler;  
    }  
}
```

```

public class Licz implements Function {

    @Override
    public double xTrue (double x){ //tutaj całkę
        double xTrue= -0.5*Math.pow(x,4) + 4*Math.pow(x,3)- 10*Math.pow(x,2) +
8.5*x + 1; //rownanie z listy 6
        //double xTrue= -0.0000797121*Math.pow(x, 21/5)-
0.0611845*Math.pow(x,3)+9.81*x; //skoczek
        //double xTrue= Math.exp(1/3*Math.pow(x,3)-1.1*x); // zad.5
        return xTrue;
    }

    @Override
    public double ft(double x, double t){
        double ft= -2*Math.pow(t,3)+ 12*Math.pow(t,2) -20*t+ 8.5;
        //double ft=9.81-
12.5/68.1*(Math.pow(t,2)+8.3/(Math.pow(46,2.2))*Math.pow(t,3.2));
        //double ft=x*Math.pow(t,2)-1.1*x;
        return ft;
    }
}

```

```

import java.util.ArrayList;

public interface Function {

    //ArrayList liczEuler(double przedzialL, double przedzialU, double krok);
    double xTrue(double t);
    double ft(double x, double t);
}

```

2.1)

Klasa umożliwiająca całkowanie równań różniczkowych zwyczajnych pierwszego rzędu zmodyfikowaną metodą Eulera:

```

import java.util.ArrayList;

public class ZmodyfikowanyEuler {

    Licz licz= new Licz();

    public ArrayList liczEuler(double przedzialL, double przedzialU, double krok){
        ArrayList<Double> listXEuler = new ArrayList();
        ArrayList<Double> listXTrue = new ArrayList();
        ArrayList<Double> listT = new ArrayList();

        double tSrodkowy;
        double wartoscSrodkowa;
        double t=przedzialL;
        double xEulerNext=0;
        double xTrue;
    }
}

```

```

        double xEulerPrevious=1; //==0
        do{
            tSrodkowy= (t+przedzialU)/2;
            listT.add(t);
            xTrue=licz.xTrue(t);
            listXTrue.add(xTrue);
            xEulerNext = Math.abs(xEulerPrevious + licz.ft(xEulerNext,
tSrodkowy)*krok); //x(0.5)
            listXEuler.add(xEulerNext);
            xEulerPrevious=xEulerNext;
            t+=krok;
        }while(t<przedzialU);

        System.out.println("ListXTrue: ");
        System.out.println(listXTrue);
        System.out.println("t: ");
        System.out.println(listT);
        return listXEuler;
    }

}

```

3.1)

```

public class Licz implements Function {

    @Override
    public double xTrue (double x){ //tutaj całkę
        double xTrue= -0.5*Math.pow(x,4) + 4*Math.pow(x,3)- 10*Math.pow(x,2) +
8.5*x + 1; //rownanie z listy 6
        //double xTrue= -0.0000797121*Math.pow(x, 21/5)-
0.0611845*Math.pow(x,3)+9.81*x; //skoczek
        //double xTrue= Math.exp(1/3*Math.pow(x,3)-1.1*x); // zad.5
        return xTrue;
    }

    @Override
    public double ft(double x, double t){
        double ft= -2*Math.pow(t,3)+ 12*Math.pow(t,2) -20*t+ 8.5;
        //double ft=9.81-
12.5/68.1*(Math.pow(t,2)+8.3/(Math.pow(46,2.2))*Math.pow(t,3.2));
        //double ft=x*Math.pow(t,2)-1.1*x;
        return ft;
    }
}
import java.util.ArrayList;

public class Test {

    static Euler euler = new Euler();
    static ZmodyfikowanyEuler zmodyfikowanyEuler = new ZmodyfikowanyEuler();
    static ArrayList<Double> listXEuler = new ArrayList();
    static ArrayList<Double> listXZmodyfikowanyEuler = new ArrayList();

```

```

public static void main(String[] args) {

    listXEuler = euler.liczEuler(0,4,0.5);
    listXZmodyfikowanyEuler = zmodyfikowanyEuler.liczEuler(0,4,0.5);
    System.out.println("Euler: ");
    System.out.println(listXEuler);
    System.out.println("Zmodyfikowany Euler: ");
    System.out.println(listXZmodyfikowanyEuler);
}
}

```

Indefinite integral:

$$\int \left(9.81 - \frac{12.5 \left(v^2 + \frac{8.3 v^{3.2}}{46^{2.2}} \right)}{68.1} \right) dv =$$

$$-0.0000797121 v^{21/5} - 0.0611845 v^3 + 9.81 v + \text{constant}$$

4.1)

```
double xEulerPrevious=licz.xTrue(0);
```

```
public class Licz implements Function {
```

```

    @Override
    public double xTrue (double x){ //tutaj całkę
        //double xTrue= -0.5*Math.pow(x,4) + 4*Math.pow(x,3)- 10*Math.pow(x,2) +
        8.5*x + 1; //rownanie z listy 6
        double xTrue= -0.0000797121*Math.pow(x, 21/5)-
        0.0611845*Math.pow(x,3)+9.81*x; //skoczek
        //double xTrue= Math.exp(1/3*Math.pow(x,3)-1.1*x); // zad.5
        return xTrue;
    }

    @Override
    public double ft(double x, double t){
        //double ft= -2*Math.pow(t,3)+ 12*Math.pow(t,2) -20*t+ 8.5;
        double ft=9.81-
        12.5/68.1*(Math.pow(t,2)+8.3/(Math.pow(46,2.2))*Math.pow(t,3.2));
        //double ft=x*Math.pow(t,2)-1.1*x;
        return ft;
    }
}

```

5.1a)

```

(double xEulerNext=1;
double xEulerPrevious=0;)

```

```

public class Licz implements Function {

    @Override
    public double xTrue (double x){ //tutaj całkę
        //double xTrue= -0.5*Math.pow(x,4) + 4*Math.pow(x,3)- 10*Math.pow(x,2) +
        8.5*x + 1; //rownanie z listy 6
        //double xTrue= -0.0000797121*Math.pow(x, 21/5)-
        0.0611845*Math.pow(x,3)+9.81*x; //skoczek
        double xTrue= Math.exp(1/3*Math.pow(x,3)-1.1*x); // zad.5
        return xTrue;
    }

    @Override
    public double ft(double x, double t){
        //double ft= -2*Math.pow(t,3)+ 12*Math.pow(t,2) -20*t+ 8.5;
        //double ft=9.81-
        12.5/68.1*(Math.pow(x,2)+8.3/(Math.pow(46,2.2))*Math.pow(x,3.2));
        double ft=x*Math.pow(t,2)-1.1*x;
        return ft;
    }
}

```

3.2)

ListXTrue:

[1.0, 3.21875, 3.0, 2.21875, 2.0, 2.71875, 4.0, 4.71875]

t:

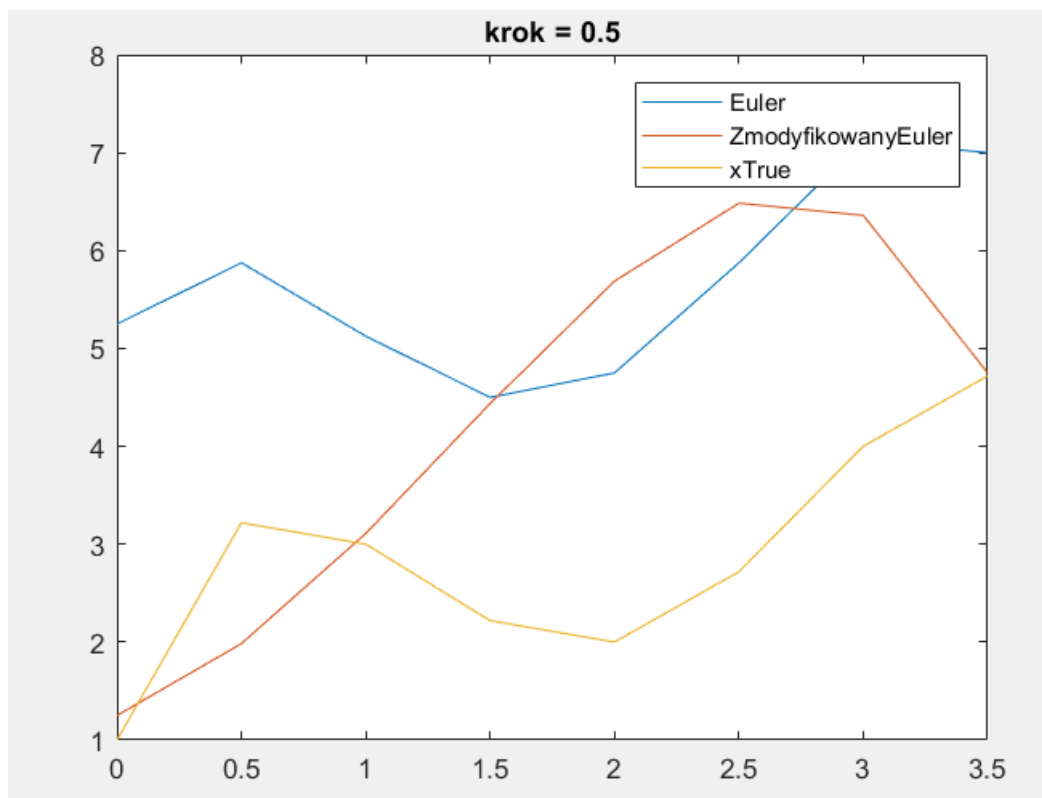
[0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5]

Euler:

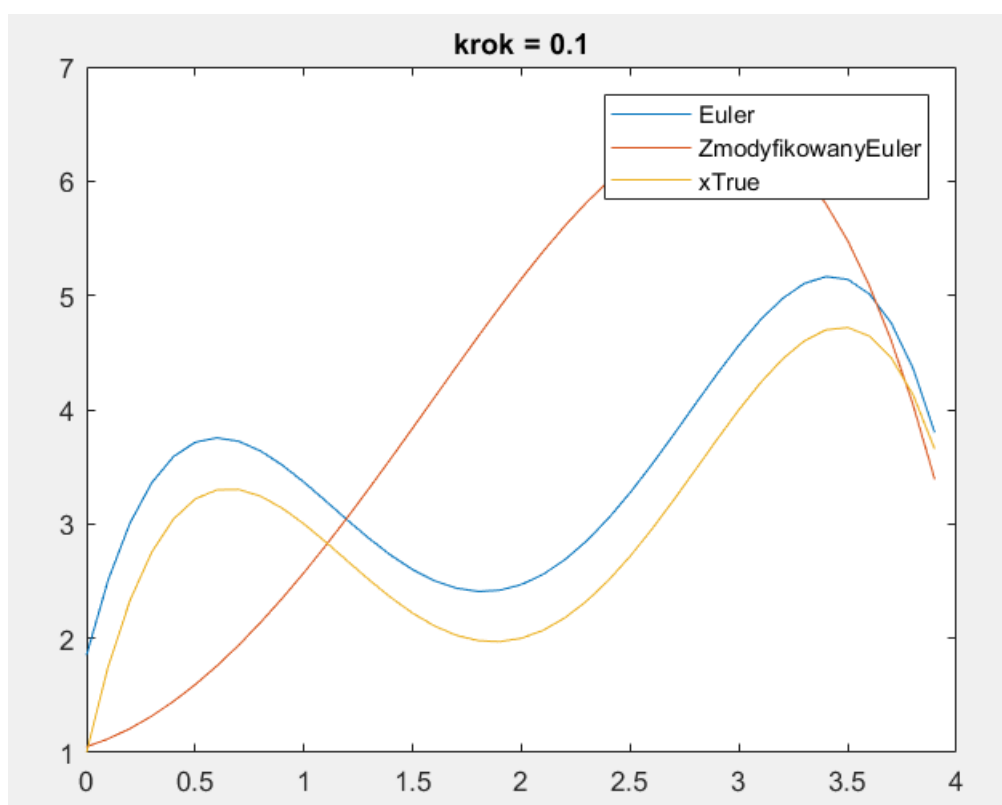
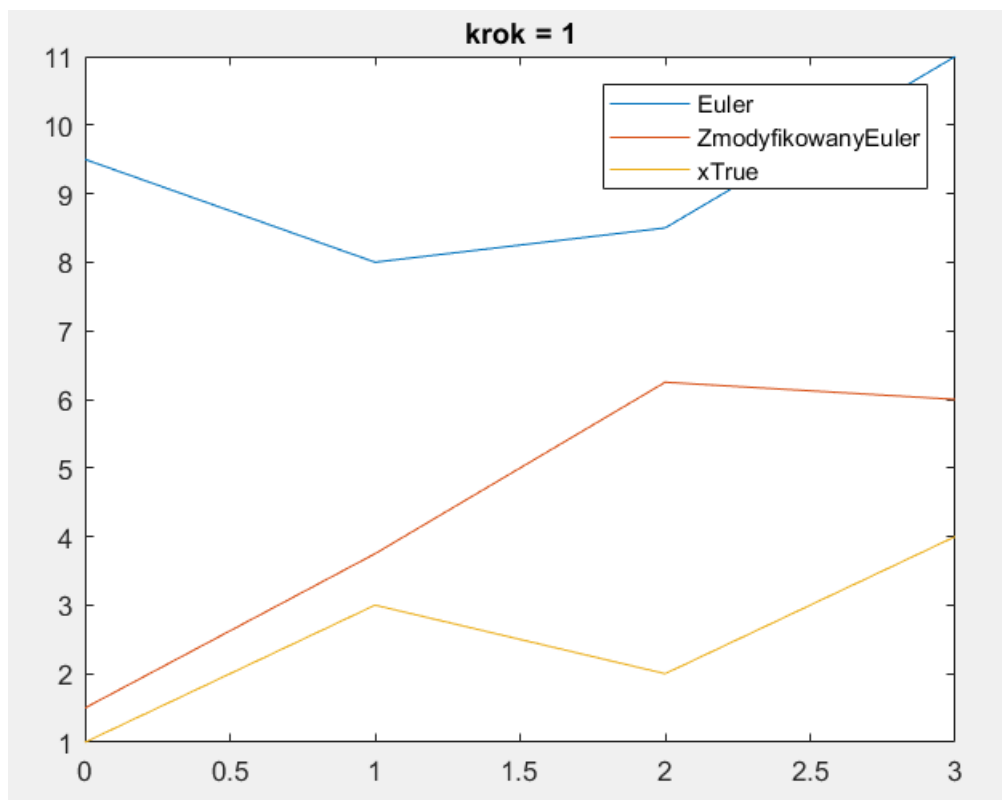
[5.25, 5.875, 5.125, 4.5, 4.75, 5.875, 7.125, 7.0]

Zmodyfikowany Euler:

[1.25, 1.984375, 3.109375, 4.4375, 5.6875, 6.484375, 6.359375, 4.75]

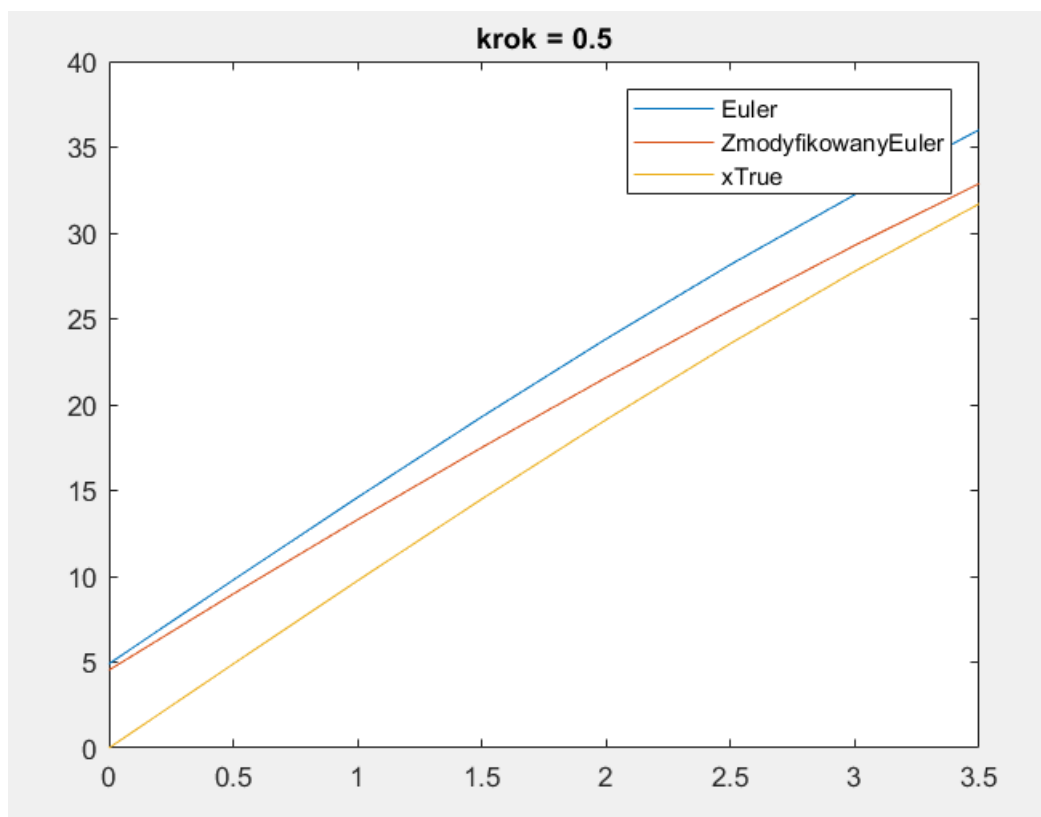


```
ListXTrue:
[0.0, 4.8973469554937505, 9.748735787900001, 14.50809876999375, 19.1292486064, 23.565878433593753, 27.7715618199, 31.69975276549375]
ListXTrue:
[0.0, 4.8973469554937505, 9.748735787900001, 14.50809876999375, 19.1292486064, 23.565878433593753, 27.7715618199, 31.69975276549375]
t:
[0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5]
Euler:
[4.905, 9.787037584513287, 14.600093390192196, 19.2979829098821, 23.83433741913549, 28.162590819912257, 32.235969327280515, 36.007482892249364]
Zmodyfikowany Euler:
[4.536354509253389, 8.97449198908517, 13.302745389861936, 17.509421398373238, 21.5827999057415, 25.511133514602577, 29.282647079571426, 32.88553727631854]
```



4.2) skoczek

KROK 0.5:



ListXTrue:

```
[0.0, 4.8973469554937505, 9.748735787900001, 14.50809876999375, 19.1292486064, 23.565878433593753, 27.7715618199, 31.69975276549375]
```

t:

```
[0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5]
```

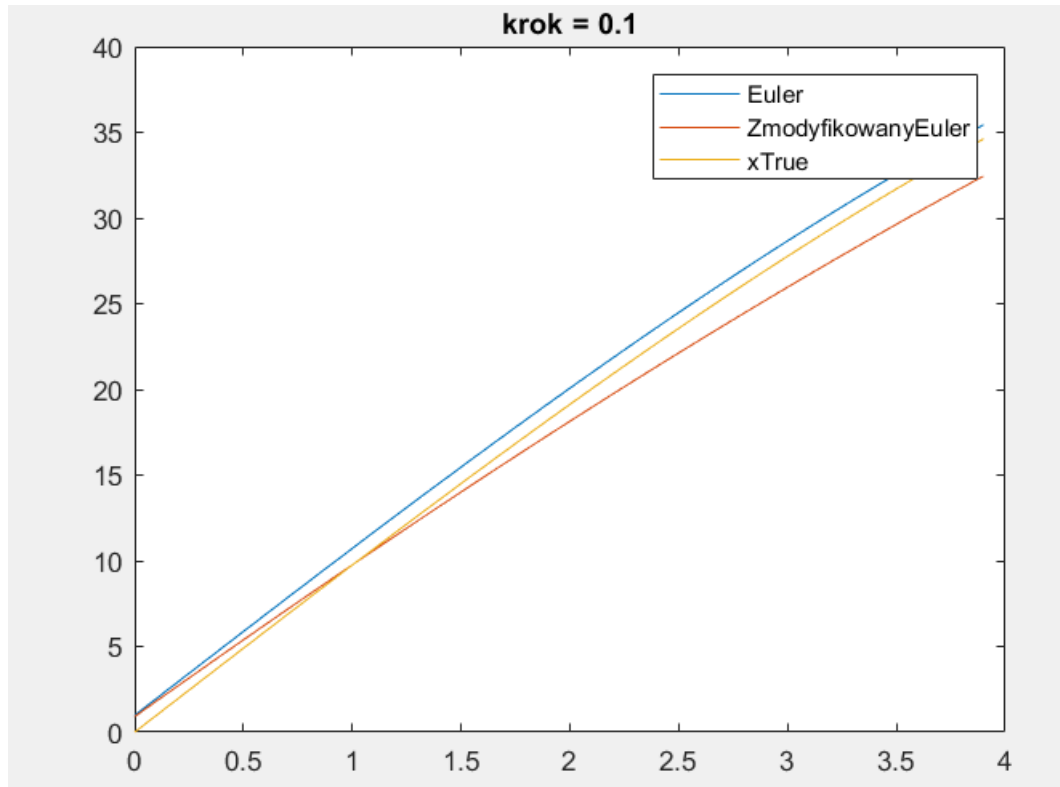
Euler:

```
[4.905, 9.787037584513287, 14.600093390192196, 19.2979829098821, 23.83433741913549, 28.162590819912257, 32.235969327280515, 36.007482892249364]
```

Zmodyfikowany Euler:

```
[4.536354509253389, 8.97449198908517, 13.302745389861936, 17.509421398373238, 21.5827999057415, 25.511133514602577, 29.282647079571426, 32.88553727631854]
```


KROK 0.1:



skoczek krok 0.1:

```
ListXTrue:
[0.0, 0.9809388075287901, 1.9615103964606402, 2.9413473728319905, 3.92008215137024, 4.8973469554937505, 5.87277381731184, 6.84599457762479, 7.81664088592384, 8.784344200391189, 9.7487357879,
10.709446724014388, 11.66610789298944, 12.618349987771191, 13.565803509996641, 14.508098769993753, 15.444865886781443, 16.375734788069593, 17.300335210259046, 18.218296698441595, 19
.129248606400004, 20.032820096607995, 20.928640140230247, 21.8163375171224, 22.69554081583105, 23.56587843359376, 24.42697857633905, 25.2784692586864, 26.11997830394625, 26.95113334412,
27.77156181990001, 28.580890980669604, 29.378747884503056, 30.164759398165607, 30.938552197113456, 31.69975276549377, 32.447987396144654, 33.1828821905952, 33.90406305906546, 34.61115572046641]

t:
[0.0, 0.1, 0.2, 0.30000000000000004, 0.4, 0.5, 0.6, 0.7, 0.7999999999999999, 0.8999999999999999, 0.9999999999999999, 1.0999999999999999, 1.2, 1.3, 1.4000000000000001, 1.5000000000000002,
1.6000000000000003, 1.7000000000000004, 1.8000000000000005, 1.9000000000000006, 2.0000000000000007, 2.1000000000000008, 2.2000000000000009, 2.300000000000001, 2.4000000000000011, 2
.5000000000000012, 2.6000000000000013, 2.7000000000000014, 2.8000000000000015, 2.9000000000000016, 3.0000000000000017, 3.1000000000000018, 3.2000000000000019, 3.300000000000002, 3.4000000000000021,
3.5000000000000022, 3.6000000000000023, 3.7000000000000024, 3.8000000000000025, 3.9000000000000026, 4.0000000000000027, 4.1000000000000028, 4.2000000000000029, 4.300000000000003, 4.4000000000000031,
4.5000000000000032, 4.6000000000000033, 4.7000000000000034, 4.8000000000000035, 4.9000000000000036, 5.0000000000000037, 5.1000000000000038, 5.2000000000000039, 5.300000000000004, 5.4000000000000041,
5.5000000000000042, 5.6000000000000043, 5.7000000000000044, 5.8000000000000045, 5.9000000000000046, 6.0000000000000047, 6.1000000000000048, 6.2000000000000049, 6.300000000000005, 6.4000000000000051,
6.5000000000000052, 6.6000000000000053, 6.7000000000000054, 6.8000000000000055, 6.9000000000000056, 7.0000000000000057, 7.1000000000000058, 7.2000000000000059, 7.300000000000006, 7.4000000000000061,
7.5000000000000062, 7.6000000000000063, 7.7000000000000064, 7.8000000000000065, 7.9000000000000066, 8.0000000000000067, 8.1000000000000068, 8.2000000000000069, 8.300000000000007, 8.4000000000000071,
8.5000000000000072, 8.6000000000000073, 8.7000000000000074, 8.8000000000000075, 8.9000000000000076, 9.0000000000000077, 9.1000000000000078, 9.2000000000000079, 9.300000000000008, 9.4000000000000081,
9.5000000000000082, 9.6000000000000083, 9.7000000000000084, 9.8000000000000085, 9.9000000000000086, 10.0000000000000087, 10.1000000000000088, 10.2000000000000089, 10.300000000000009, 10.4000000000000091,
10.5000000000000092, 10.6000000000000093, 10.7000000000000094, 10.8000000000000095, 10.9000000000000096, 11.0000000000000097, 11.1000000000000098, 11.2000000000000099, 11.30000000000001, 11.400000000000011,
11.500000000000012, 11.600000000000013, 11.700000000000014, 11.800000000000015, 11.900000000000016, 12.000000000000017, 12.100000000000018, 12.200000000000019, 12.30000000000002, 12.400000000000021,
12.500000000000022, 12.600000000000023, 12.700000000000024, 12.800000000000025, 12.900000000000026, 13.000000000000027, 13.100000000000028, 13.200000000000029, 13.30000000000003, 13.400000000000031,
13.500000000000032, 13.600000000000033, 13.700000000000034, 13.800000000000035, 13.900000000000036, 14.000000000000037, 14.100000000000038, 14.200000000000039, 14.30000000000004, 14.400000000000041,
14.500000000000042, 14.600000000000043, 14.700000000000044, 14.800000000000045, 14.900000000000046, 15.000000000000047, 15.100000000000048, 15.200000000000049, 15.30000000000005, 15.400000000000051,
15.500000000000052, 15.600000000000053, 15.700000000000054, 15.800000000000055, 15.900000000000056, 16.000000000000057, 16.100000000000058, 16.200000000000059, 16.30000000000006, 16.400000000000061,
16.500000000000062, 16.600000000000063, 16.700000000000064, 16.800000000000065, 16.900000000000066, 17.000000000000067, 17.100000000000068, 17.200000000000069, 17.30000000000007, 17.400000000000071,
17.500000000000072, 17.600000000000073, 17.700000000000074, 17.800000000000075, 17.900000000000076, 18.000000000000077, 18.100000000000078, 18.200000000000079, 18.30000000000008, 18.400000000000081,
18.500000000000082, 18.600000000000083, 18.700000000000084, 18.800000000000085, 18.900000000000086, 19.000000000000087, 19.100000000000088, 19.200000000000089, 19.30000000000009, 19.400000000000091,
19.500000000000092, 19.600000000000093, 19.700000000000094, 19.800000000000095, 19.900000000000096, 20.000000000000097, 20.100000000000098, 20.200000000000099, 20.3000000000001, 20.40000000000011,
20.50000000000012, 20.60000000000013, 20.70000000000014, 20.80000000000015, 20.90000000000016, 21.00000000000017, 21.10000000000018, 21.20000000000019, 21.3000000000002, 21.40000000000021,
21.50000000000022, 21.60000000000023, 21.70000000000024, 21.80000000000025, 21.90000000000026, 22.00000000000027, 22.10000000000028, 22.20000000000029, 22.3000000000003, 22.40000000000031,
22.50000000000032, 22.60000000000033, 22.70000000000034, 22.80000000000035, 22.90000000000036, 23.00000000000037, 23.10000000000038, 23.20000000000039, 23.3000000000004, 23.40000000000041,
23.50000000000042, 23.60000000000043, 23.70000000000044, 23.80000000000045, 23.90000000000046, 24.00000000000047, 24.10000000000048, 24.20000000000049, 24.3000000000005, 24.40000000000051,
24.50000000000052, 24.60000000000053, 24.70000000000054, 24.80000000000055, 24.90000000000056, 25.00000000000057, 25.10000000000058, 25.20000000000059, 25.3000000000006, 25.40000000000061,
25.50000000000062, 25.60000000000063, 25.70000000000064, 25.80000000000065, 25.90000000000066, 26.00000000000067, 26.10000000000068, 26.20000000000069, 26.3000000000007, 26.40000000000071,
26.50000000000072, 26.60000000000073, 26.70000000000074, 26.80000000000075, 26.90000000000076, 27.00000000000077, 27.10000000000078, 27.20000000000079, 27.3000000000008, 27.40000000000081,
27.50000000000082, 27.60000000000083, 27.70000000000084, 27.80000000000085, 27.90000000000086, 28.00000000000087, 28.10000000000088, 28.20000000000089, 28.3000000000009, 28.40000000000091,
28.50000000000092, 28.60000000000093, 28.70000000000094, 28.80000000000095, 28.90000000000096, 29.00000000000097, 29.10000000000098, 29.20000000000099, 29.300000000001, 29.4000000000011,
29.5000000000012, 29.6000000000013, 29.7000000000014, 29.8000000000015, 29.9000000000016, 30.0000000000017, 30.1000000000018, 30.2000000000019, 30.300000000002, 30.4000000000021, 30.5000000000022,
30.6000000000023, 30.7000000000024, 30.8000000000025, 30.9000000000026, 31.0000000000027, 31.1000000000028, 31.2000000000029, 31.300000000003, 31.4000000000031, 31.5000000000032, 31.6000000000033,
31.7000000000034, 31.8000000000035, 31.9000000000036, 32.0000000000037, 32.1000000000038, 32.2000000000039, 32.300000000004, 32.4000000000041, 32.5000000000042, 32.6000000000043, 32.7000000000044,
32.8000000000045, 32.9000000000046, 33.0000000000047, 33.1000000000048, 33.2000000000049, 33.300000000005, 33.4000000000051, 33.5000000000052, 33.6000000000053, 33.7000000000054, 33.8000000000055,
33.9000000000056, 34.0000000000057, 34.1000000000058, 34.2000000000059, 34.300000000006, 34.4000000000061, 34.5000000000062, 34.6000000000063, 34.7000000000064, 34.8000000000065, 34.9000000000066,
35.0000000000067, 35.1000000000068, 35.2000000000069, 35.300000000007, 35.4000000000071, 35.5000000000072, 35.6000000000073, 35.7000000000074, 35.8000000000075, 35.9000000000076, 36.0000000000077,
36.1000000000078, 36.2000000000079, 36.300000000008, 36.4000000000081, 36.5000000000082, 36.6000000000083, 36.7000000000084, 36.8000000000085, 36.9000000000086, 37.0000000000087, 37.1000000000088,
37.2000000000089, 37.300000000009, 37.4000000000091, 37.5000000000092, 37.6000000000093, 37.7000000000094, 37.8000000000095, 37.9000000000096, 38.0000000000097, 38.1000000000098, 38.2000000000099,
38.30000000001, 38.400000000011, 38.500000000012, 38.600000000013, 38.700000000014, 38.800000000015, 38.900000000016, 39.000000000017, 39.100000000018, 39.200000000019, 39.30000000002, 39.400000000021,
39.500000000022, 39.600000000023, 39.700000000024, 39.800000000025, 39.900000000026, 40.000000000027, 40.100000000028, 40.200000000029, 40.30000000003, 40.400000000031, 40.500000000032, 40.600000000033,
40.700000000034, 40.800000000035, 40.900000000036, 41.000000000037, 41.100000000038, 41.200000000039, 41.30000000004, 41.400000000041, 41.500000000042, 41.600000000043, 41.700000000044, 41.800000000045,
41.900000000046, 42.000000000047, 42.100000000048, 42.200000000049, 42.30000000005, 42.400000000051, 42.500000000052, 42.600000000053, 42.700000000054, 42.800000000055, 42.900000000056, 43.000000000057,
43.100000000058, 43.200000000059, 43.30000000006, 43.400000000061, 43.500000000062, 43.600000000063, 43.700000000064, 43.800000000065, 43.900000000066, 44.000000000067, 44.100000000068, 44.200000000069,
44.30000000007, 44.400000000071, 44.500000000072, 44.600000000073, 44.700000000074, 44.800000000075, 44.900000000076, 45.000000000077, 45.100000000078, 45.200000000079, 45.30000000008, 45.400000000081,
45.500000000082, 45.600000000083, 45.700000000084, 45.800000000085, 45.900000000086, 46.000000000087, 46.100000000088, 46.200000000089, 46.30000000009, 46.400000000091, 46.500000000092, 46.600000000093,
46.700000000094, 46.800000000095, 46.900000000096, 47.000000000097, 47.100000000098, 47.200000000099, 47.3000000001, 47.40000000011, 47.50000000012, 47.60000000013, 47.70000000014, 47.80000000015,
47.90000000016, 48.00000000017, 48.10000000018, 48.20000000019, 48.3000000002, 48.40000000021, 48.50000000022, 48.60000000023, 48.70000000024, 48.80000000025, 48.90000000026, 49.00000000027, 49.10000000028,
49.20000000029, 49.3000000003, 49.40000000031, 49.50000000032, 49.60000000033, 49.70000000034, 49.80000000035, 49.90000000036, 50.00000000037, 50.10000000038, 50.20000000039, 50.3000000004, 50.40000000041,
50.50000000042, 50.60000000043, 50.70000000044, 50.80000000045, 50.90000000046, 51.00000000047, 51.10000000048, 51.20000000049, 51.3000000005, 51.40000000051, 51.50000000052, 51.60000000053, 51.70000000054,
51.80000000055, 51.90000000056, 52.00000000057, 52.10000000058, 52.20000000059, 52.3000000006, 52.40000000061, 52.50000000062, 52.60000000063, 52.70000000064, 52.80000000065, 52.90000000066, 53.00000000067,
53.10000000068, 53.20000000069, 53.3000000007, 53.40000000071, 53.50000000072, 53.60000000073, 53.70000000074, 53.80000000075, 53.90000000076, 54.00000000077, 54.10000000078, 54.20000000079, 54.3000000008,
54.40000000081, 54.50000000082, 54.60000000083, 54.70000000084, 54.80000000085, 54.90000000086, 55.00000000087, 55.10000000088, 55.20000000089, 55.3000000009, 55.40000000091, 55.50000000092, 55.60000000093,
55.70000000094, 55.80000000095, 55.90000000096, 56.00000000097, 56.10000000098, 56.20000000099, 56.300000001, 56.4000000011, 56.5000000012, 56.6000000013, 56.7000000014, 56.8000000015, 56.9000000016,
57.0000000017, 57.1000000018, 57.2000000019, 57.300000002, 57.4000000021, 57.5000000022, 57.6000000023, 57.7000000024, 57.8000000025, 57.9000000026, 58.0000000027, 58.1000000028, 58.2000000029, 58.300000003,
58.4000000031, 58.5000000032, 58.6000000033, 58.7000000034, 58.8000000035, 58.900000
```

$$\ln x = 1/3 \cdot x^3 - 1,1x + C$$

$$x = e^{(1/3 \cdot x^3 - 1,1x + C)}$$

$$x = e^{(1/3 \cdot x^3 - 1,1x)} + C$$

5.2b)

Krok 0.5:

```
ListXTrue:
[1.0, 0.5769498103804866, 0.33287108369807955, 0.19204990862075408]
ListXTrue:
[1.0, 0.5769498103804866, 0.33287108369807955, 0.19204990862075408]
t:
[0.0, 0.5, 1.0, 1.5]
Euler:
[0.55, 0.31625000000000003, 0.3004375, 0.47318906250000003]
Zmodyfikowany Euler:
[0.050000000000000044, 0.06156250000000005, 0.09696093750000007, 0.19210385742187513]
```

Krok 0.25:

```
ListXTrue:
[1.0, 0.7595721232249685, 0.5769498103804866, 0.4382349924649492, 0.33287108369807955, 0.25283959580474646,
0.19204990862075408, 0.14587575685622733]
ListXTrue:
[1.0, 0.7595721232249685, 0.5769498103804866, 0.4382349924649492, 0.33287108369807955, 0.25283959580474646,
0.19204990862075408, 0.14587575685622733]
t:
[0.0, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75]
Euler:
[0.275, 0.203671875, 0.16039160156249999, 0.13883898010253903, 0.13536800559997555, 0.15101993124747273, 0.
.19443816148112114, 0.2898343844577962]
Zmodyfikowany Euler:
[0.025000000000000022, 0.02603515625000002, 0.029045471191406273, 0.03478649010658267, 0.04478760601222519,
0.06203783239037129, 0.0924751439068972, 0.14832146128192186]
```