Решение задачи Коши.

Некоторые методы численного решения задачи Коши для дифференциальных уравнений.

Требуется определить сеточную функцию y_i отвечающую задаче:

$$\dot{y} \equiv \frac{dy}{dx} = f(x,y) \qquad y(x_0) = y_0$$

Метод Эйлера: $y_{i+1} = y_i + h*f(x_i,y_i)$

Один из методов Рунге-Кутта 2-го порядка точности

$y_{i+1} = y_i + \Delta y_i$, $\Delta y_i = 0.5*h*(k_1 + k_2)$,

$k_1 = f(x_i,y_i)$, $k_2 = f(x_i + h, y_i + k_1)$.

Один из методов Рунге-Кутта 4-го порядка точности

$y_{i+1} = y_i + \Delta y_i$, $\Delta y_i = (1/6)*h*(k_1 + 2*k_2 + 2*k_3 + 4*k_4)$,

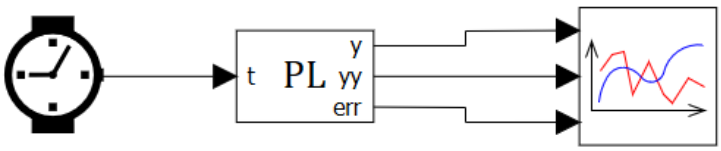
$k_1 = f(x_i,y_i)$, $k_2 = f(x_i + h/2, y_i + (h/2)* k_1)$,

$k_3 = f(x_i + h/2, y_i + (h/2)* k_2)$, $k_4 = f(x_i + h, y_i + h* k_3)$,

При численном решении дифференциальные уравнения заменяются разностными. Решение полученных разностных уравнений может оказаться неустойчивым, хотя исходная система обыкновенных дифференциальных уравнений была устойчивой. Неустойчивость проявляется как катастрофический рост ошибки численного решения при увеличении размера шага. Покажем на примере решения задачи Коши

$$\dot{y} \equiv \frac{dy}{dt} = 50 * (e^{-t} - y) \qquad y(t = 0\ ) = 1$$

В SimInTech



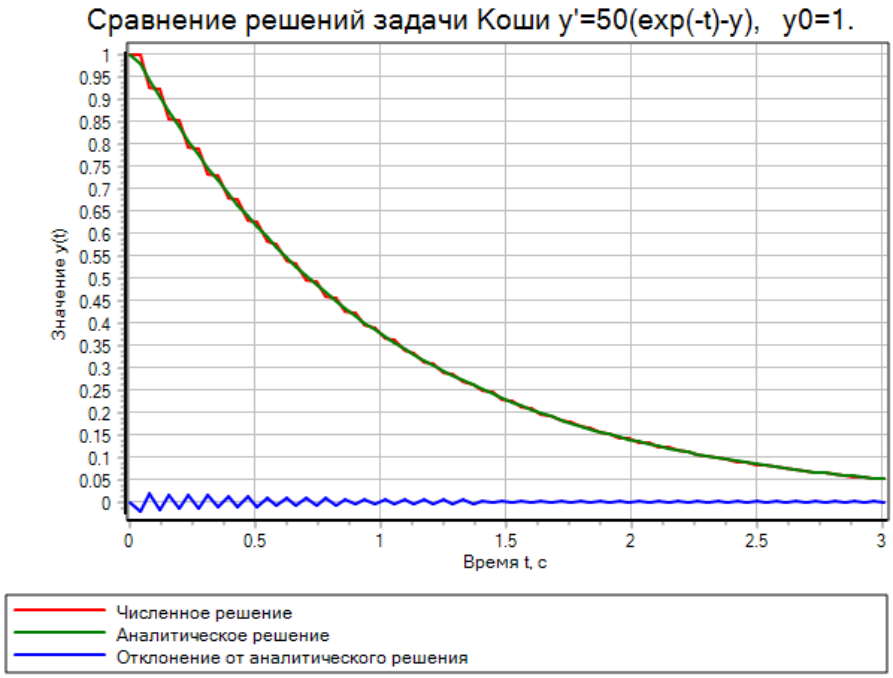
Скрипт проекта

```
// Решение задачи Коши  (250227)
initialization
        var
        h       = 0.039,   // Шаг интегрирования м
        t_fin   = 3.0,     // Конечное время моделирования, с
        y0    = 1.0;       // Начальное значение искомой функции (при t = 0 )
end;
```

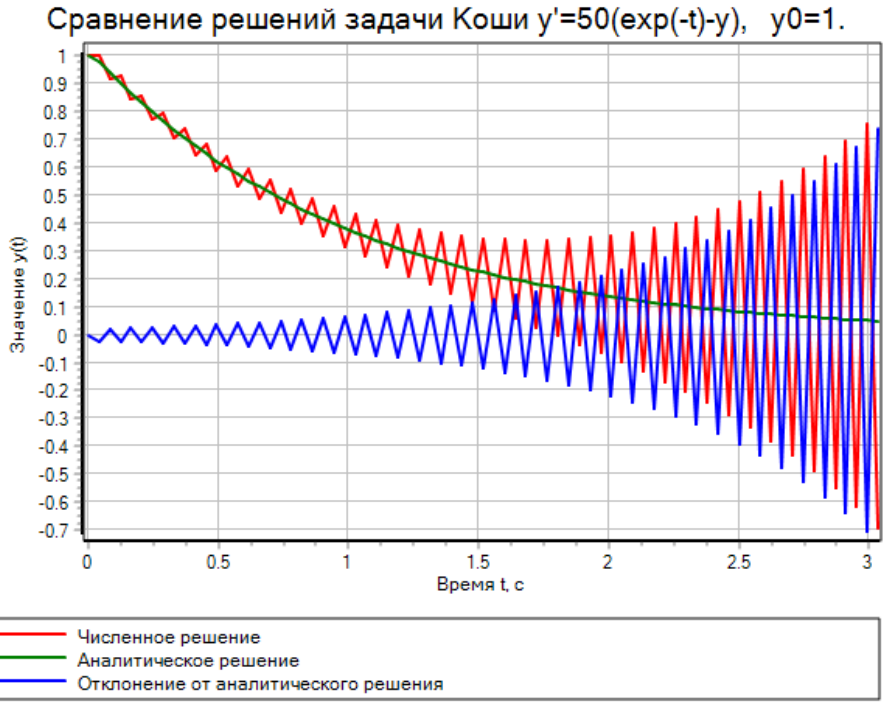
Программа

```
// Задача Kochi 250227
input t;
init z=1;
output y, yy, err;   // Численное и аналитическое решения, разность решений
// Аналитическое решение
function yy_of(t: double):double
        yy_of = (50.0*exp(-t)- exp(-50.0*t))/49.0;
end;
z' = 50*(exp(-t) - z);
if goodstep then begin y = z;   yy = yy_of(time); err = yy - y;  end;
```

При шаге интегрирования h= 0.039 решение –



При шаге интегрирования h= 0.041 решение



То есть численное решение не устойчиво. Обычно для конкретной задачи и конкретного метода существует некоторое граничное значение шага h_{max} , превышение которого приводит к неустойчивости численного решения.

Задача № 1 (радиоактивный распад)
Радиоактивный распад описывается уравнением

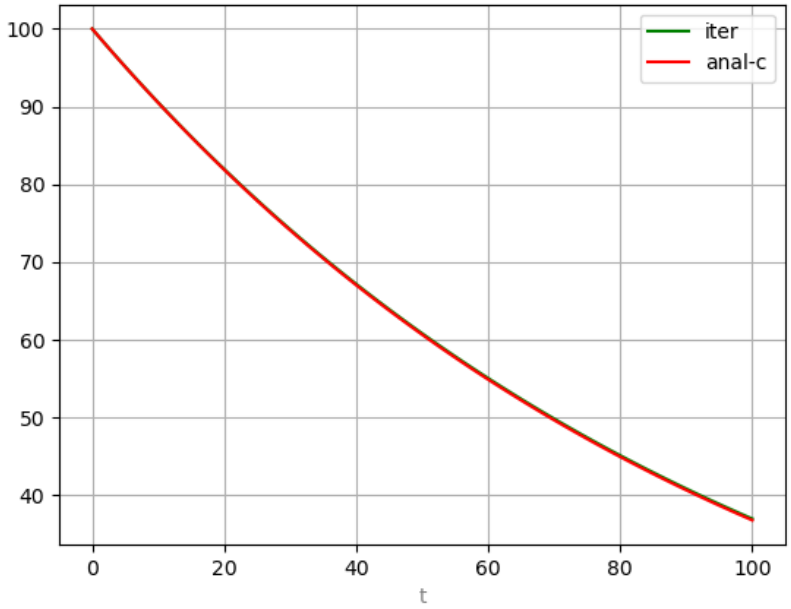
$$y' = -ky$$

Сколько вещества останется в момент времени t = 100, если k = 0.01, а в начальный момент времени y(t=0) = 100 г.
Сравнить результат с аналитическим решением y = 100*exp(-kt).
Используем метод Эйлера -

Программа на Python –

```
import math
import numpy as np
import matplotlib.pyplot as plt
def funct(t):
    global k
    y = 100*math.exp(-k*t)
    return y
k = 0.01
tfin = 100
n = 100
tay = tfin/(n)
t = np.linspace(0, tfin, n)
yy = np.zeros(n)   # yy = 100*exp(-kt)
yy[0] = 100
y = np.zeros(n)
y[0] = 100
for i in range(n-1):
    y[i+1] = y[i] *(1 - tay*k)
    yy[i+1]= funct(t[i+1])
print("y(t=100) =", y[n-1])
plt.xlabel('t', color='gray')
plt.plot(t,y,color="green", label='iter')
plt.plot(t,yy,color="red", label='anal-c')
plt.legend()
plt.grid(True)
plt.show()
plt.title('Massa dy(x)/dt = -k*y(t) ', fontsize=20, fontname='Times New Roman')
```

```
Результат работы программ
y(t=100) = 36.97296376497267
```



Программа на C –

```
#include <iostream>
#include <string>
#include <math.h>
using namespace std;
int main()
{
	double kk = 0.01;
	double ya = 0;
	double yii = 0;
	double yi = 100;
	double timefinish = 100.0;
	int  n = 100;
	double tt = 0;
	double tay = timefinish /n ;
	cout << "Culculation	Analitic	Delta	" << endl;
	for (int k = 0; k < 10; k++) {
	for (int i = 0; i < 10; i++)
	{   tt+=tay;
		yii = yi - kk * yi * tay;
		ya = 100.0 * exp(-kk * tt);
		yi = yii;
	}
	  cout << "  "<< yii<<   "  " << ya  <<   "   " << fabs(yii - ya) << endl;
	}
	cout << "  Finis time  = " <<  tt << "   Massa  = "<<   yi <<endl;
  }
```

Программа на Java –

```
package eiy;
public class eee {
	public static void main(String[] args) {
	double k=0.01, ya=0, yii=0,yi=100, timefinis= 100;
	int n= 10000;
	double tay= timefinis/(n-1.0);
	System.out.println("  time "+ " yi    "+  "  ya" + "      |yi-ya|" );
```

```
        for(int i=0; i<n; i++){
        yii = yi-k*yi*tay;
        ya=100*Math.exp(-k*tay*(i+1));
//      System.out.println(tay*(i-1)+"  "+ yii+ "  "+ ya + " " + Math.abs(yii-ya));
        yi=yii;
        }
        System.out.println("!!!  "+tay*(n-1)+"  "+ yii+ "  "+ ya + " " + Math.abs(yii-
ya));
        }

}
```

Решение в SimInTech -

Представим решение задачи радиоактивного распада с использованием среды динамического моделирования технических систем SimInTech.

Скрипт проекта (проект в схеме модели общего вида) -

```
// Решение задачи радиоактивного распада y' = -ky   (250222)
initialization
        var  k          = 0.01,         // Постоянная радиоакт.  распада грамм/с
             t_fin      = 100.0,        //Конечное время моделирования, с
             y0         = 100.0;        //Начальное значение массы (при t = 0 ), грамм
end;
```

Проект в схеме модели общего вида (Слой «Автоматика»)-

Блок PL на языке программирования (язык программирования библиотеки «Динамические») -

```
        // Задача радиоактивного распада 250224
        init z=y0;
        output y, err;   // y - решение, err - отклонение от аналитического решения
        // Аналитическое решение
        function yy_of(t: double):double
                yy_of = y0*exp(-k*t);
        end;
        z' = -k*z
        if goodstep then begin y = z;   err = yy_of(time) - y; end;
```

Вкладка «Параметры расчета»

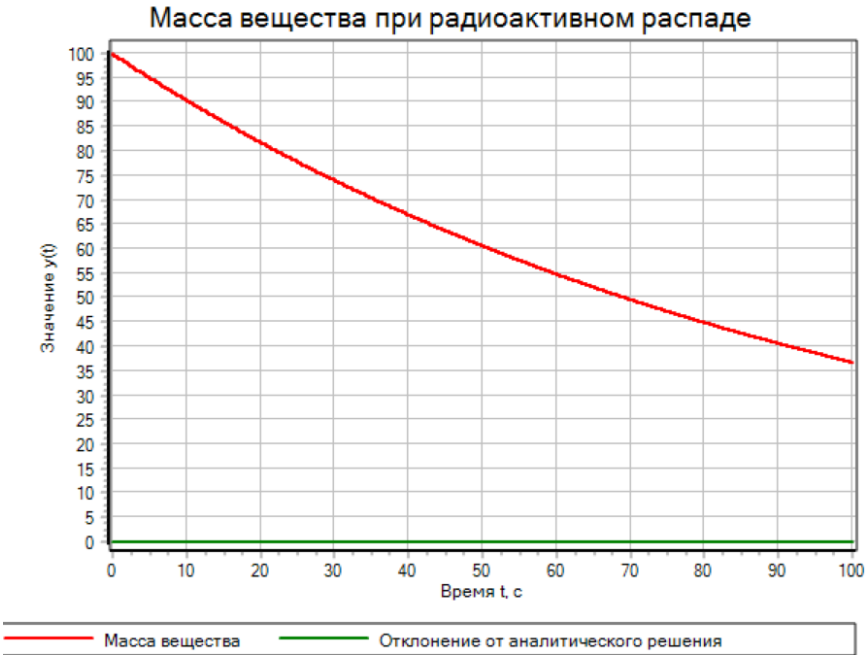
Параметры проекта: E:\Document-20241016\SimInTech-Учеба(250214)\Work-240224\Task-250224-II.prt слой: Автома...

Параметры расчёта | Управление расчётом | Настройки проекта

Название	Имя	Формула	Значение
Основные параметры			
Минимальный шаг	hmin		0.0001
Максимальный шаг	hmax		0.0001
Шаг синхронизации задачи в пакете	synstep		0
Режим выполнения задачи в пакете	serial_mode		Параллельный
Начальный шаг интегрирования (если 0 - выбирается автоматически)	startstep		0
Метод интегрирования	intmet		Эйлера
Начальное время расчёта	starttime		0
Конечное время расчёта	endtime	t_fin	100
Относительная ошибка	relerr		0.0001
Абсолютная ошибка	abserr		1E-6
Относительная ошибка сравнения времени для дискретных блоков и источников	time_rel_error		1E-12
Начальное значение неинициализированных выходов блоков	InitOutputsValue		0
Генерация кода			
Управление расчётом			
Настройки решения НАУ			
Визуализация данных			
Удалённая отладка кода			
Сортировка блоков			
Тонкие настройки решения СЛАУ			
Тонкие настройки решения НАУ			
Электрические схемы			

Редактировать список параметров

Результаты моделирования –



Задача № 2

Тело с начальной массой m = 200 кг ускоряется постоянной силой в F = 2000 ньютонов. Масса тела уменьшается на dm = 1 кг/сек. Сила сопротивления воздуха Fb пропорциональна скорости тела Fb = K*V с коэффициентом пропорциональности K = 2 н*сек/м.

Найти скорость тела через 50 сек, если в момент времени t = 0 тело находилось в покое.

РЕШЕНИЕ

Дифференциальное уравнение записывается в виде –

$$\frac{dV}{dt} = \frac{1}{m - dm * t} * (F - K * V),$$

$$\frac{dV}{dt} = \frac{1}{200 - t} * (200 - 2 * V),$$

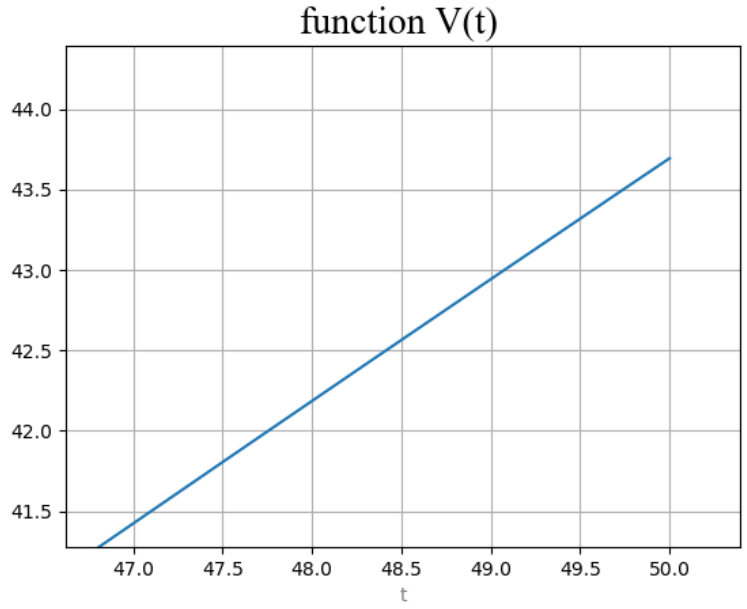
Граничное условие V(t=0) = 0.

Программа (Python) [программа с ошибкой в записи уравнения]

```
import math
import numpy as np
import matplotlib.pyplot as plt
n     = 500
timef = 50
tay = (timef - 0)/n
time = 0
V = 0
for i in range(n):
    time = time + tay
    Vnew = V+tay*(200 - 2*V)/(200 - time)
    V = Vnew
print("V(t=",time,") = ", V)
t = np.linspace(0, timef, n)
y = np.zeros(n)
y[0] = 0
for i in range(n-1):
    y[i+1] = y[i]+tay*(200 - 2*y[i])/(200 - t[i])
plt.title('function V(t) ', fontsize=20, fontname='Times New Roman')
plt.xlabel('t', color='gray')
plt.plot(t,y)
plt.grid(True)
plt.show()
```

Результат:

V(t= 50.00000000000044) = 43.77814532892075



Программа C [программа с ошибкой в записи уравнения]

```
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
	double vi = 0.0;
	double vii = 0.0;
	double time = 0.0;
	double time_mode = 50.0;
	double tay = 0.1;
	int n = (int)(time_mode / tay);
	for (int i = 0; i < n; i++) {
		vii = vi + tay * (200 - 2 * vi) / (200 - time);
		time += tay;
		vi = vii;
	}
	cout << "vv V(" << time << ") = " << vii << endl;
}
```

Программа Java [программа с ошибкой в записи уравнения]

```
package prim1;
import java.text.DecimalFormat;
public class prim {
	public static void main(String[] args) {
//		dV/dt = (200-2*V)/(200-t)	V(t=0) = 0
	DecimalFormat ft = new DecimalFormat("#0.0");
	DecimalFormat fv = new DecimalFormat("#0.000");
	double vi=0, vii=0, time=0, time_mod=50, tay=0.1 ;
```

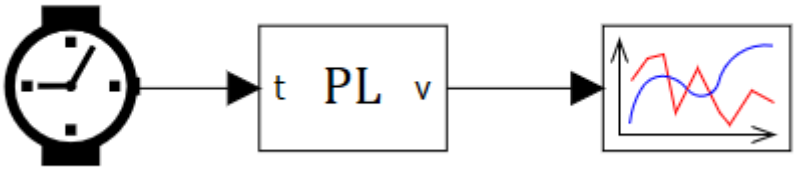
```
        int n = (int) (time_mod/tay);

        for(int i=0; i<n; i++){
                vii=vi+tay*(200-2*vi)/(200-time);
                time+=tay;
                vi=vii;
        }
        System.out.println("V("+ft.format(time)+")="+fv.format(vii));
        }
}
```

Ответ:

```
V(50,0)=43,759
```

Решение в SimInTech –

Проект в схеме модели общего вида (Слой «Автоматика»)-



Скрипт проекта (проект в схеме модели общего вида) -

```
// Задача - Скорость тела переменной массы   (250225)
initialization
        var
                V0              = 0,            // Начальная скорость м/с
                m0              = 200,          // Начальная масса тела, кг
                dm              = 1,            // Коэфф. изменения массы, кг/с
                F               = 200,          // Тяга двигателя, н
                K               = 2,            // Коэфф. сопротивления воздуха, н*сек/м
                t_fin           = 50.0;         // Конечное время моделирования, с
end;
```

Блок PL на языке программирования (язык программирования библиотеки «Динамические») -

```
        // Задача - Скорость тела переменной массы   (250225)
        input t;
        init z=V0;
        output V;   // Решение
        z' = (F-K*z)/(m0 - dm*time)
        if goodstep then begin V = z;  end;
```

Результаты моделирования

Задача № 3

Материальная точка массой m

```
package eiy;

public class eee {

	public static void main(String[] args) {
	double L=0.1, Qi=3.1415/4, T=2*3.1415*Math.sqrt(L/9.81);
	System.out.println("  T= "+T );
	double Uii=0, Ui=0, Qii=0;
	int n= 1000;
	double tay= T/(n-1.0);
	System.out.println("  time "+ " yi     " );
	for(int i=0; i<n; i++){
	Uii = Ui-tay*(9.8*Math.sin(Qi)/L);
	Qii = Qi + tay*Ui;
	System.out.println(tay*(i-1)+"  "+ Qii     );
	Ui = Uii;
	Qi = Qii;
	}
		}

}
```

Задача № 4

Рассмотрим простую экосистему, состоящую из кроликов, для которых имеется неограниченный запас пищи, и лис, которые для пропитания охотятся за кроликами. Простейшая математическая модель системы (модель Вольтерра) представляется в виде:

$$\frac{dr}{dt} = 2r - \alpha * rf, \qquad r(0) = r_0,$$
$$\frac{df}{dt} = -f + \alpha * rf, \quad f(0) = f_0.$$

где t – время; r = r(t) – число кроликов; f = f(t) – число лис; α = const > 0. При α > 0 лисы встречают кроликов с вероятностью, пропорциональной произведению числа тех и других.

Исследуйте поведение этой системы для α = 0,01 и различных значений r_0 и f_0 т. нескольких единиц до нескольких тысяч:

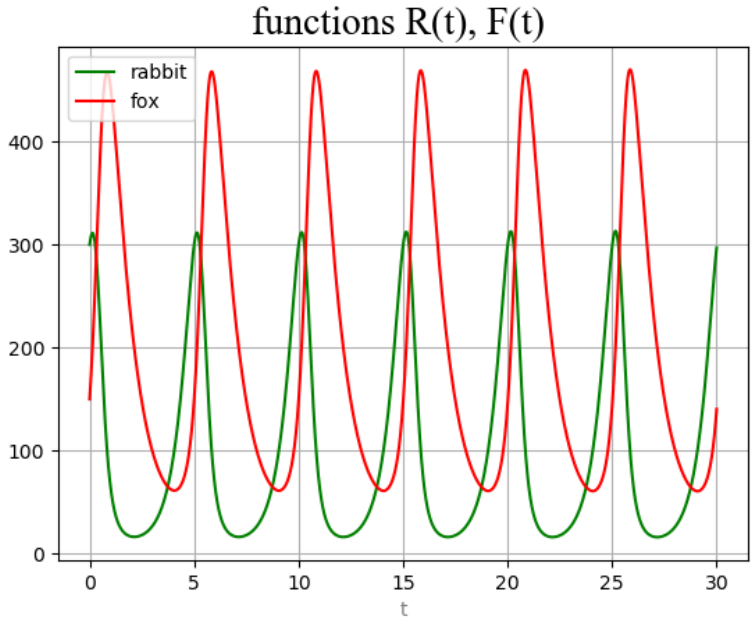
а) r_0 = 300, f_0 = 150,

б) r_0 = 15, $f_0 = 22$.

Программа (Python)

```
import math
import numpy as np
import matplotlib.pyplot as plt
n     = 100000
timef = 30
tay = (timef - 0)/n
time = 0
R00 = 300
F00 = 150
R0 = R00
F0 = F00
alfa = 0.01

i = 0
while (time < timef):
    i += 1
    time = time + tay
    Rnew = R0 + tay*(2*R0 - alfa* R0* F0)
    Fnew = F0 + tay*(-F0 + alfa* R0* F0)
    R0 = Rnew
    F0 = Fnew
    if i % 1000 == 0 :
        print(R0, '  ', F0)
t = np.linspace(0, timef, n)
R = np.zeros(n)
F = np.zeros(n)
R[0] = R00
F[0] = F00
for i in range(n-1):
    R[i+1] = R[i]+tay*(2*R[i] - alfa* R[i]* F[i])
    F[i+1] = F[i]+tay*(-F[i] + alfa* R[i]* F[i])
plt.title('functions R(t), F(t) ', fontsize=20, fontname='Times New Roman')
```

```
plt.xlabel('t', color='gray')
plt.plot(t,R,color="green", label='rabbit')
plt.plot(t,F,color="red", label='fox')
plt.legend()
plt.grid(True)
plt.show()
```



Программа (C)

```
#include <iostream>
#include <math.h>


using namespace std;

int main()
	double r = 150.0; //Так же можно заменить значение для выполнения условия задачи под буквой "Б" r = 15.0
	double rii = 0.0;
	double f = 300.0; //Так же можно заменить значение для выполнения условия задачи под буквой "Б" f = 22.0
	double fii = 0.0;
	double alfa = 0.01;
	double tay = 0.01;
	double time = 5.0;
	int n = (int)(time / tay);
	double* kr_m = new double[n];
	double* li_m = new double[n];
	double* time_m = new double[n];
	kr_m[0] = r;
	li_m[0] = f;
	time_m[0] = 0;
```

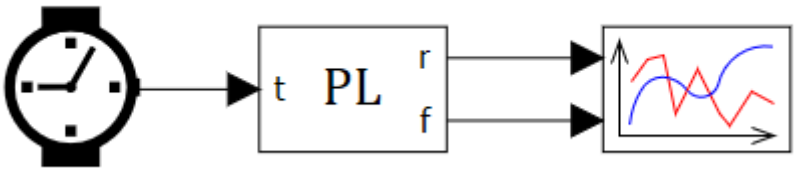
```
	int kk = 20;
	for (int i = 0; i < n; i++)
	{
		kr_m[i + 1] = kr_m[i] + tay * (2 * kr_m[i] - alfa * kr_m[i] * li_m[i]);
		li_m[i + 1] = li_m[i] + tay * (-li_m[i] + alfa * kr_m[i] * li_m[i]);
		time_m[i + 1] = time_m[i] + tay;
		if ((i % kk) == 0)
		{
			cout << i << " h\t tt = " << time_m[i + 1] << "\t  KK = " << kr_m[i + 1] <<
"\t  LL = " << li_m[i + 1] << endl;
		}
	}
}
```

Программа (Java)

```
public class kk {

	public static void main(String[] args) {
	double r = 300, rii = 0;
	double f = 150, fii = 0;
	double alfa = 0.01;
	double tay = 0.01, time = 5;
	int n = (int) (time/tay);
	double [] kr_m = new double [n];
	double [] li_m = new double [n];
	double [] time_m = new double [n];
	kr_m [0] = r;
	li_m [0] = f;
	time_m [0] =0;
	int kk = 20;
	for (int i=0; i< n; i++){
		kr_m [i+1] = kr_m [i]+tay*(2*kr_m [i]-alfa*
				kr_m [i]*li_m[i]);
		li_m [i+1] = li_m [i]+tay*(-li_m [i]+ alfa*
				kr_m [i]*li_m[i]);
		time_m [i+1] = time_m [i] +tay;
    if  ( (i % kk) ==0)   System.out.println ("time="+time_m[i+1]  +"  krr =
"+kr_m[i+1] +
				"  lis = "+ li_m[i+1]);
	}
	}
}
```

Решение в SimInTech –

Проект в схеме модели общего вида (Слой «Автоматика»)-



Скрипт проекта (проект в схеме модели общего вида) -

```
// Задача - Скорость тела переменной массы   (250225)
initialization
        var
                R0              = 300,          // rabbit,          unit
                F0              = 150,          // fox, unit
                alfa            = 0.01,         // alfa
                t_fin           = 30.0;         // Конечное время моделирования
 end;
```

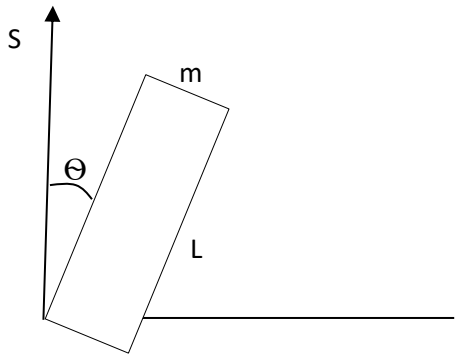
Блок PL на языке программирования (язык программирования библиотеки «Динамические») -

```
// Задача - модель Вольтерра  (250225)
input t;
init zr=R0, zf=F0;
output r, f;   // Решение
zr' = 2*zr - alfa*zr*zf
zf' = -zf  + alfa*zr*zf
if goodstep then begin r = zr; f = zf  end;
```

Результаты моделирования

Задача № 5

Посредством электромотора вынуждаются быстрые колебания поддерживающего шарика перевернутого маятника в вертикальном направлении: $S = Asin\omega * t$.



Подготовить программу для моделирования движения перевернутого маятника. С помощью программы исследовать движение маятника при следующих значениях параметров системы L, A, ω и начальных значениях Θ (0), $\dot{\Theta}$(0):

L, (м)	A, (м)	ω, (рад/сек)	Θ (0), (рад)	$\dot{\Theta}$(0), (рад/сек)
0,25	0	0	3,1	0
0,25	0,012	5,3	3,1	0
0,25	0,25	100	3,1	0
0,25	0,25	100	0,1	0
0,25	0,05	100	0,1	0
0,25	0,012	200	0,05	0

Применение второго закона Ньютона дает уравнение движения:

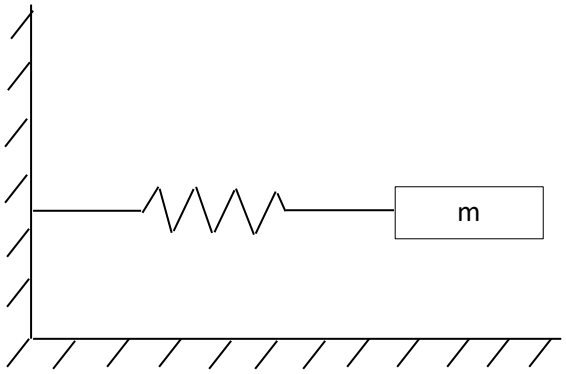
$$\ddot{\theta} = \frac{3}{2*L} * (g - A * \omega^2 sin\omega t) sin\theta,$$

где g – ускорение силы тяжести. Для малых значений Θ (sin Θ ≈ Θ) уравнение превращается в уравнение Матье, которое устойчиво для некоторых значений A, ω и начальных условий. Движение перевернутого маятника будет устойчиво лишь в определенной области.

Задача №6

Тело движется по плоской поверхности с трением, обуславливающим демпфирование колебаний. Его масса m = 45кг, жесткость пружин К = 175 н/м, коэффициент трения f = 0,3. Рассчитать движение тела в интервале времени 0 ≤ t ≤ 2 сек при начальных условиях: x(0) = 7,5см, x(0) = 0.

Результаты представить графически.



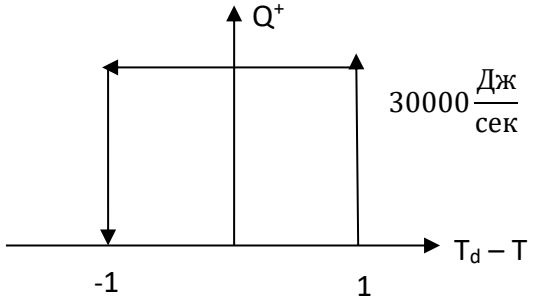
Задача №7

Система отопления некоторого объекта описывается уравнением

$$Q^+ - Q^- = 250\, dT \;/dt,$$

где Q^+ - подвод тепла от нагревателя, Q^- - потери тепла в окружающую среду,

T – температура объекта.



Объект оборудован датчиком, который включает и выключает нагреватели в зависимости от разницы температуры объекта T и заданной температуры T_d так, чтобы разница составляла не более одного градуса. Включение нагревателя соответствует графику, представленному на рисунке.

Пусть Q^- = 500 T , заданная температура T_d= 20°C, а начальная температура объекта $T_0 = 10°C$.

Методами математического моделирования исследовать систему отопления объекта на предмет эффективности системы.

Под эффективностью работы системы E будем понимать величину численно равную отношению времен $t^- / (t^- + t^+)$, где t^+ и t^- - время действия нагревателя и время отключения нагревателя, соответственно.

Решение:

Имеем задачу Коши

$\frac{dT}{dt} = \frac{1}{250} * (Q^+ - Q^-)$, $T(0) = T_0$

Простейшая схема -

$T_{i+1} = T_i + \tau/250 * (Q^+ - Q^-) = T_i + \tau/250 * (Q^+ - 500 * T_i)$,

Программа для начального исследования –

```
//    https://www.programiz.com/cpp-programming/online-compiler/
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
  double Td = 20, T0 = 10, k_p = 500, dT = 1.0, Qp = 30000.0;
   double T = T0, TT, Q=Qp;
   double time_mod = 0, time_finis= 600.0;
   double time_upr = 0;                    // Время действия нагревателя
```

```
   double time_upr_neg =0;              //  Время отключения нагрева
   int num_print = 20, num_mod = 10000;       // Кол-во печати, кол-во проходов по циклу
   double tay = (time_finis/(num_print*num_mod));
   cout<<" Время,  "<< "  T,    "<< "       Время "<< "       Время отключения  "<<"\n";
   cout<<"  c  "<< "    град. C  "<< "       нагрева,c"<< "         нагрева,c"<<"\n";
   for (int k=0; k<num_print; k++){    // Цикл печати
          for (int i=0; i<num_mod; i++)
                { time_mod += tay;                // Моделируемое время
                 if ((T<Td)) Q =Qp;  else Q=0 ;
                 TT    = T + tay*(Q - k_p*T)/250.0;
                 if (Q > 0) time_upr += tay; else time_upr_neg += tay;
                 T     = TT;
                 }
        cout<<time_mod<<"      "<< T<<"         "<<time_upr<< "          "<<time_upr_neg<<" \n
";
    }
    double Eff;
    Eff = time_upr_neg / (time_upr_neg  + time_upr  );
     cout <<" \n "<<      "  Eff =  "<<Eff << "     tay =  "<< tay<<"\n ";      //  Эффективность
    return 0;
}}
```

Время, c	T, град. C	Время нагрева,с	Время отключения нагрева,с
30	19.8817	10.083	19.917
60	20.0147	20.094	39.906
90	20.2358	30.105	59.895
120	20.0015	40.113	79.887
150	20.1342	50.124	99.876
180	19.9857	60.132	119.868
210	20.1221	70.143	139.857
240	19.8923	80.151	159.849
270	20.0972	90.162	179.838
300	19.8813	100.17	199.83
330	20.0113	110.181	219.819
360	20.21	120.192	239.808
390	20.0011	130.2	259.8
420	20.1311	140.211	279.789
450	19.9621	150.219	299.781
480	20.1217	160.23	319.77
510	19.8895	170.238	339.762
540	20.0754	180.249	359.751
570	19.8809	190.257	379.743
600	20.0087	200.268	399.732

Eff = 0.66582   tay = 0.006   N = 100000

Программа для исследования эффективности работы нагревателя в зависимости от точности удерживаемой температуры и времени моделирования процесса.

```
#include <iostream>
#include <cmath>
```

```
using namespace std;
int main()
{
  double Td = 20, T0 = 10, k_p = 500, dT = 0.5, Qp = 30000.0;
   double T = T0, TT, Q=Qp;
   double time_mod = 0, time_finis= 600.0;
   double time_upr = 0;              // Время действия нагревателя
   double time_upr_neg =0;       //  Время отключения нагрева
   int num_print = 20;               // Кол-во печати
   double tay = 2*dT*250/(Qp-500*Td);
   int num_mod = (int) (time_finis/tay);
   cout<<" Время,  "<< "  T,    "<< "       Время "<< "       Время отключения  "<<"\n";
   cout<<"  c  "<< "    град. C  "<< "       нагрева,c"<< "         нагрева,c"<<"\n";
   int num_mod_pr = (int)(1.0*num_mod/num_print);
   for (int k=0; k<num_print; k++){    // Цикл печати
            for (int i=0; i<num_mod_pr; i++)
                    { time_mod += tay;               // Моделируемое время
                     if ((T<Td)) Q =Qp;  else Q=0 ;
                     TT    = T + tay*(Q - k_p*T)/250.0;
                     if (Q > 0) time_upr += tay; else time_upr_neg += tay;
                     T     = TT;
                    }
     cout<<time_mod<<"     "<< T<<"         "<<time_upr<< "          "<<time_upr_neg<<" \n ";
     }
     double Eff;
     Eff = time_upr_neg / (time_upr_neg  + time_upr  );
     cout <<" \n "<<     "  Eff =  "<<Eff << "    tay = "<< tay<<"     N = "<<num_mod<< " \n ";     //
Эффективность
     return 0;
}
```

Время, c	T, град. C	Время нагрева,c	Время отключения нагрева,c
30	20.0114	10.125	19.875
60	19.6242	20.175	39.825
90	19.5042	30.225	59.775
120	20.5766	40.2875	79.7125
150	20.5129	50.3375	99.6625
180	20.0319	60.3875	119.613
210	19.7919	70.4375	139.563
240	19.5151	80.4875	159.512
270	20.6657	90.55	179.45
300	20.5186	100.6	199.4
330	20.0792	110.65	219.35
360	20.0011	120.7	239.3
390	19.5402	130.75	259.25
420	20.871	140.813	279.187
450	20.532	150.862	299.137
480	20.1883	160.912	319.087
510	20.0082	170.962	339.037
540	19.5981	181.012	358.987
570	19.5025	191.062	378.937
600	20.5627	201.125	398.875

Eff = 0.664792 tay = 0.0125 N = 48000

Для задачи Коши

$\frac{dT}{dt} = \frac{1}{250} * (Q^+ - Q^-)$, T(0) = T_0

используем схему второго порядка точности с введением дополнительной сеточной функции Td_i

$Td_i = T_i + \tau/250 * (Q^+ - Q^-) = T_i + \tau/250 * (Q^+ - 500 * T_i)$,

$T_{i+1} = T_i + \frac{\tau}{250} * 0.5 * ((Q^+ - 500 * T_i) + (Q^+ - 500 * Td_i))$,

Программа для исследования эффективности работы нагревателя в зависимости от точности удерживаемой температуры и времени моделирования процесса.

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
  double Td = 20, T0 = 10, k_p = 500, dT = 0.5, Qp = 30000.0;
   double T = T0, TT, Q=Qp, Tb;
   double time_mod = 0, time_finis= 600.0;
   double time_upr = 0;                  // Время действия нагревателя
   double time_upr_neg =0;               //  Время отключения нагрева
   int num_print = 20;                   // Кол-во печати
   double tay = 2*dT*250/(Qp-500*Td);
   int num_mod = (int) (time_finis/tay);// Кол-во проходов по циклу
   cout<<" Время,  "<< "  T,    "<< "       Время "<< "       Время отключения  "<<"\n";
   cout<<"  c  "<< "    град. C  "<< "       нагрева,c"<< "        нагрева,c"<<"\n";
   int num_mod_pr = (int)(1.0*num_mod/num_print);
   for (int k=0; k<num_print; k++){    // Цикл печати
          for (int i=0; i<num_mod_pr; i++)
                { time_mod += tay;              // Моделируемое время
                 if ((T<Td)) Q =Qp;  else Q=0 ;
          Tb    = T + tay*(Q - k_p*T)/250.0;
          TT    = T + (tay/250)*0.5*((Q - k_p*T)+ (Q - k_p*Tb) );
                 if (Q > 0) time_upr += tay; else time_upr_neg += tay;
                  T     = TT;
                 }
     cout<<time_mod<<"      "<< T<<"          "<<time_upr<< "          "<<time_upr_neg<<" \n ";
     }
     double Eff;
     Eff = time_upr_neg / (time_upr_neg  + time_upr  );
     cout <<" \n "<<      "  Eff =  "<<Eff << "    tay = "<< tay<<"     N = "<<num_mod<< " \n ";
//  Эффективность
     return 0;
}
```

Время, c	T, град. C	Время нагрева,c	Время отключения нагрева,c
30	20.0109	10.125	19.875
60	19.8195	20.175	39.825
90	19.5546	30.225	59.775

120	19.5109	40.275	79.725
150	20.685	50.3375	99.6625
180	20.5322	60.3875	119.613
210	20.507	70.4375	139.563
240	20.1015	80.4875	159.512
270	20.0133	90.5375	179.462
300	19.8708	100.588	199.412
330	19.5631	110.638	219.362
360	19.5123	120.688	239.312
390	20.7147	130.75	259.25
420	20.5371	140.8	279.2
450	20.5078	150.85	299.15
480	20.1185	160.9	319.1
510	20.0161	170.95	339.05
540	19.9304	181	359
570	19.573	191.05	378.95
600	19.5139	201.1	398.9

Eff = 0.664833 tay = 0.0125 N = 48000

Для задачи Коши

$\frac{dT}{dt} = \frac{1}{250} * (Q^+ - Q^-)$, T(0) = T_0

используем схему четвертого порядка точности с введением дополнительных функций k1, k2, k3, k4.

Решение строится на основе схемы Рунге-Кутта четвертого порядка точности-

$T_{i+1} = T_i + (1/6)*\tau*(k_1 + 2*k_2 + 2*k_3 + k_4)$,

$k_1 = f(t_i,T_i)$, $k_2 = f(t_i + \tau/2, T_i + (\tau/2)* k_1)$,

$k_3 = f(t_i + \tau/2, T_i + (\tau/2)* k_2)$, $k_4 = f(t_i + \tau, T_i + \tau * k_3)$,

Для данной задачи, имеем

k1 = $(Q^+ - 500 * T_i)/250$

k2 = $(Q^+ - 500 * (T_i + 0.5 * \tau * k1)/250$

k3 = $(Q^+ - 500 * (T_i + 0.5 * \tau * k2)/250$

k2 = $(Q^+ - 500 * (T_i + \tau * k3)/250$

$T_{i+1} = T_i + (1/6)*(\tau)*(k_1 + 2*k_2 + 2*k_3 + k_4)$,

,

Программа для исследования эффективности работы нагревателя в зависимости от точности удерживаемой температуры и времени моделирования процесса.

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
   double Td = 20, T0 = 10, k_p = 500, dT = 0.5, Qp = 30000.0;
    double T = T0, TT, Q=Qp;
    double k1,k2,k3,k4;
    double time_mod = 0, time_finis= 600.0;
```

```
    double time_upr = 0;                    // Время действия нагревателя
    double time_upr_neg =0;                 //  Время отключения нагрева
    int num_print = 20; //, num_mod = 100000;        // Кол-во печати,
    double tay = 2*dT*250/(Qp-500*Td);
    int num_mod = (int) (time_finis/tay);// Кол-во проходов по циклу
    cout<<" Время,  "<< "  T,    "<< "        Время "<< "       Время отключения  "<<"\n";
    cout<<"  c  "<< "    град. C  "<< "       нагрева,c"<< "         нагрева,c"<<"\n";
    int num_mod_pr = (int)(1.0*num_mod/num_print);
    for (int k=0; k<num_print; k++){    // Цикл печати
            for (int i=0; i<num_mod_pr; i++)
                    { time_mod += tay;                  // Моделируемое время
                     if ((T<Td)) Q =Qp;  else Q=0 ;
                     k1 = (Q - k_p*T)/250.0;
                     k2 = (Q - k_p*(T+0.5*tay*k1))/250.0;
                     k3 = (Q - k_p*(T+0.5*tay*k2))/250.0;
                     k4 = (Q - k_p*(T+tay*k3))/250.0;
           TT    = T + ((tay)/6)*(k1+2*k2+2*k3+k4 );
                     if (Q > 0) time_upr += tay; else time_upr_neg += tay;
                      T     = TT;

                     }
      cout<<time_mod<<"      "<< T<<"          "<<time_upr<< "           "<<time_upr_neg<<" \n ";
      }
      double Eff;
      Eff = time_upr_neg / (time_upr_neg  + time_upr  );
      cout <<" \n "<<       "  Eff =  "<<Eff << "    tay = "<< tay<<"     N = "<<num_mod<< " \n ";
//  Эффективность
      return 0;
}
```

Время, c	T, град. C	Время нагрева,c	Время отключения нагрева,c
30	20.012	10.125	19.875
60	19.8441	20.175	39.825
90	19.5587	30.225	59.775
120	19.5115	40.275	79.725
150	20.6993	50.3375	99.6625
180	20.5346	60.3875	119.613
210	20.5074	70.4375	139.563
240	20.1097	80.4875	159.512
270	20.0147	90.5375	179.462
300	19.8995	100.588	199.412
330	19.5678	110.638	219.362
360	19.513	120.688	239.312
390	20.7313	130.75	259.25
420	20.5399	140.8	279.2
450	20.5083	150.85	299.15
480	20.1281	160.9	319.1
510	20.0177	170.95	339.05
540	19.9638	181	359
570	19.5784	191.05	378.95
600	19.5147	201.1	398.9

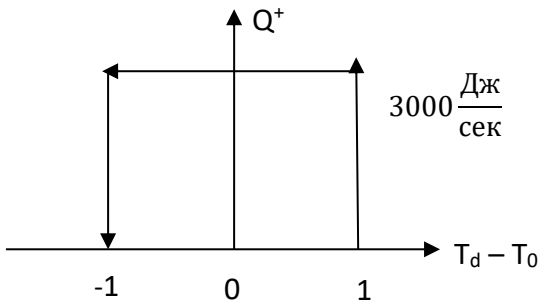
Eff = 0.664833 tay = 0.0125 N = 48000

Схема Эйлера	Eff = 0.66582	tay = 0.006	N = 100000
Второй порядок точности	Eff = 0.664833	tay = 0.0125	N = 48000
Четвертый порядок точности	Eff = 0.664833	tay = 0.0125	N = 48000

Задача № 4.7

Система отопления некоторого объекта описывается уравнением

$$Q^+ - Q^- = 250\, dT_0/dt,$$



где Q^+ - подвод тепла от нагревателя,
Q^- - потери тепла в окружающую среду,
T_0– температура объекта.

Объект оборудован датчиком, который включает и выключает нагреватели в зависимости от разницы температуры объекта и заданной температуры T_d в соответствии с представленным графиком.

Пусть Q^- = 500 T_0.

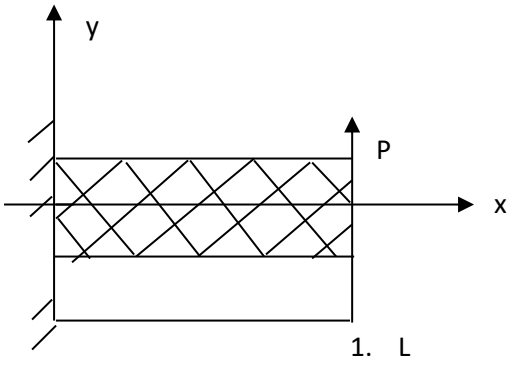
Составить программу для моделирования отопления объекта, если T_d= 20°C, а начальная температура объекта $T_0 = 10$°C.

Какова частота предельного цикла для рассматриваемой системы?

```
public static void main(String[] args) {
    double tt = 0, t = 10, td = 20;
    double qink = 3000, qk = 500, qout = 0, qin = 0;
    double time = 0, tay = 1;
    int n = 100;
    for (int i=0; i<n; i++){
      if (Math.abs(td - t) < 1)
      tt = t + (tay/250)
    }

      }
```

Задача № 4.8

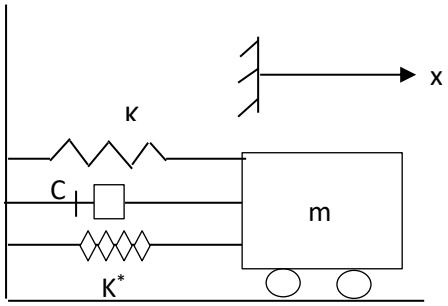
Дифференциальное уравнение изгибной линии бруска с постоянным поперечным сечением имеет вид: $\frac{y''}{(1+y'^2)^{3/2}} = \frac{P*L^2}{E}\left(\frac{1}{L} - \frac{x}{L^2}\right)$.



Составить программу для расчета изгибной линии бруска y(x), если L = 1,0 м, $PL^2/E = 2{,}0$; y(0) = 0, $y'(0) = 0$.

Задача № 9

Устройство, показанное на рисунке, состоит из массы m, связанной с жесткой стенкой через пружину постоянной жесткости К, демпфер с коэффициентом димпфирования С и пружиной с нелинейной характеристикой, создающей восстанавливающую силу, равную произведению постоянной $К^*$ на смещение в третьей степени.



Подготовить программу для моделирования движения механической системы в интервале времени $0 \leq t \leq 1$ сек. Параметры системы имеют следующие значения: К = 2.0 $^{\text{Н}}/_{\text{см}}$, $К^*$ = 2.0 $^{\text{Н}}/_{\text{сек}^2}$, С = 0.15 $^{\text{н*с}}/_{\text{см}}$, m = 1.0 кг = 0.01 $^{\text{н*с}^2}/_{\text{см}}$

Начальные условия заданы в виде: x (0) = 10см, $x'(0) = 0$.

Решение

Движение системы описывается дифференциальным уравнением $m\ddot{x} + c\dot{x} + Kx + K^*x^3 = 0$.

Сведем дифференциальное уравнение второго порядка к двум дифференциальным уравнениям первого порядка. В результате получим

$$\dot{x} = V \qquad x(0) = 10$$

$$V = -\frac{c}{m}V - \frac{K}{m}x - \frac{K^*x^3}{m} \qquad V(0) = 0$$