Одесский институт усовершенствования учителей
Одесский национальный университет им. И.И.Мечникова
Учебно-производственный центр "Интеллект"
Ришельевский лицей

П.А.Виктор

ШКОЛЬНИКУ

О КОМПЬЮТЕРНЫХ МЕТОДАХ

В ФИЗИКЕ

ПОСОБИЕ-ПРАКТИКУМ ДЛЯ УЧАЩИХСЯ
ВЫПУСКНЫХ КЛАССОВ ШКОЛ И ЛИЦЕЕВ
С УГЛУБЛЕННЫМ ИЗУЧЕНИЕМ
ФИЗИКИ И ИНФОРМАТИКИ

ОДЕССА - 2005

Книга издана на средства
гранта Президента Украины
для одаренной молодежи.

Павел Андреевич ВИКТОР

ШКОЛЬНИКУ О КОМПЬЮТЕРНЫХ МЕТОДАХ В ФИЗИКЕ

ПОСОБИЕ-ПРАКТИКУМ ДЛЯ УЧАЩИХСЯ
ВЫПУСКНЫХ КЛАССОВ ШКОЛ И ЛИЦЕЕВ
С УГЛУБЛЕННЫМ ИЗУЧЕНИЕМ
ФИЗИКИ И ИНФОРМАТИКИ

Книга содержит десятки задач из самых различных разделов физики. Их особенность в том, что для решения необходимы написание компьютерных программ, реализующих те или иные численные методы, их выполнение на компьютере и анализ полученных результатов. Постановка задач предваряется описанием типичных приемов программирования, используемых численных методов и примерами законченных программ на языке Pascal.

Для учащихся выпускных классов средних учебных заведений с углубленным изучением физико-математических дисциплин, учителей физики, информатики, для всех, кто интересуется использованием компьютеров в физике.

Введение

В основу данной книги положен опыт проведения уроков информатики в 11-х физических классах Ришельевского лицея города Одессы в течение нескольких последних лет. Она представляет собой пособие-практикум для тех, кто уже завершает углубленное изучение школьного курса физики и хотел бы научиться практическому использованию компьютера как мощнейшего средства для решения многих интересных физических задач. Книга может быть полезна и студентам младших курсов физических специальностей университетов.

Языком программирования для решения задач выбран Turbo-Pascal, ставший в последнее время "языком повседневного общения" среди тех, кто занимается программированием в школах, гимназиях, лицеях. При этом для достижения достаточно серьезных результатов требуется знание лишь самых основ языка, таких как циклы, массивы, работа с текстовыми файлами, простейшие процедуры и функции. В тех случаях, когда численный метод, необходимый для решения задачи, слишком громоздок (например, при решении систем линейных алгебраических уравнений, при реализации метода наименьших квадратов), предлагается использование "стандартных" процедур специально написанного автором пользовательского модуля Numerics, исходный текст которого имеется в приложении.

Разбиение книги на главы определяется тем, к какому классу математических объектов приводит решение той или иной физической задачи. Как правило, аналитическое решение предложенных задач невозможно. В начале глав рассматривается численный метод, используемый для решения, даются образцы работающих программ по принципу "от простого к сложному". Для лучшего освоения тех или иных численных методов иногда предлагаются

тренировочные примеры (несложные задачи в чисто математической постановке). В конце глав предлагается 8 вариантов физических задач, для решения которых требуется владение данным численным методом. Во многих случаях задачи довольно сложны, и поэтому учащимся предлагается набор "подсказок". Здесь же приводятся фрагменты таблиц, которые должны получиться в результате расчетов (ответы). Наиболее сложные задачи отмечены звездочкой.

Решение каждой задачи завершается построением графика рассчитанной зависимости. Для этого предполагается использование широко распространенной готовой программы EASYPLOT[1], строящей графики на экране компьютера по исходным данным из текстового файла. Однако графики по задачам первой главы настоятельно рекомендуется строить вручную на миллиметровой бумаге. Опыт показывает, что даже старшеклассники часто испытывают серьезные трудности, например, с выбором масштабов по координатным осям.

В заключение необходимо подчеркнуть, что целью практикума является не получение колонок цифр или кривых, а более глубокий анализ физики моделируемых явлений. Располагая компьютером, исследователь получает возможность еще с одной стороны посмотреть на изучаемый объект. Иногда именно этот взгляд является решающим для понимания того, как устроена Природа.

[1] Программу EASYPLOT и модуль Numerics Вы можете найти в INTERNET на WEB-сайте Ришельевского лицея http://www.rl.odessa.ua/Physics/programs .

Глава 1

Решение задач, сводящихся к многократным вычислениям значений функции

В практике физических исследований часто требуется рассчитать значения какой-либо довольно громоздкой функции (или нескольких функций) для большого числа значений аргумента. Язык Паскаль позволяет решать эту задачу очень простыми средствами с помощью программ, содержащих всего несколько операторов. Рассмотрим типичные примеры составления программ вычисления значений явно заданных функций.

1. Организация циклических вычислений на языке Pascal

Основные приемы программирования циклических вычислений рассмотрим на конкретном примере. Пусть требуется рассчитать координату y тела, брошенного вертикально вверх с начальной скоростью v_0 = 5 м/с. Начальная координата y_0 = 0. Расчет необходимо провести для значений времени t = 0, 0.1, 0.2, ... 1 с.

Как известно, зависимость координаты тела от времени для данного случая имеет вид

$$y = v_0 t - \frac{gt^2}{2} \qquad (1),$$

где g = 9.81 м/с^2 - ускорение свободного падения.

Для многократных вычислений по формуле (1) может быть использован любой из циклов языка Pascal:

цикл с предусловием

```
while <условие> do
  <оператор>;
```

цикл с постусловием

```
repeat
  <операторы>
until <условие>;
```

цикл с фиксированным параметром

```
for <параметр>:= <нач.значение> to <кон.значение> do
  <оператор>;
```

Воспользуемся циклом первого типа. Его выполнение начинается с проверки условия, стоящего за словом `while`. Если это условие выполняется, то выполняется оператор, стоящий за словом `do` (тело цикла), затем происходит повторная проверка условия и т.д. Если в теле цикла нужно выполнить несколько операторов, они заключаются в операторные скобки `begin ... end`.

Ниже приводится пример простейшей программы для вычисления координат вертикально брошенного тела (программа P1). Как и любая программа на языке Pascal, она состоит из двух разделов: *раздела описаний* (от слова `program` до `BEGIN`) и *раздела операторов* (от `BEGIN` до `END`.).

Раздел описаний начинается с описания констант (`const`), содержащих данные, приведенные в условии задачи: начальную скорость тела `Vo`, ускорение свободного падения `g`, время окончания вычислений `Tmax` и шаг изменения времени `Step`. Далее идет описание величин, являющихся переменными (`var`) в данной задаче - это координата тела `y` и время движения `t`. Поскольку эти величины могут принимать дробные значения, соответствующие переменные имеют тип `real`.

Раздел операторов начинается с присваивания начального нулевого значения времени. Затем, поскольку условие `t <= Tmax` выполняется, программа входит в цикл. Здесь вначале производятся вычисления по формуле (1), затем процедура `Writeln` выводит на экран значения времени и координаты. Третий оператор в теле цикла увеличивает значение времени на величину шага, подготавливая тем самым вычисления в следующий момент времени. Далее снова идет проверка условия `t<= Tmax` и вычисления повторяются. При одиннадцатом прохождения цикла оператор `t:= t + Step` увеличивает значение времени с 1.0 до 1.1, и проверка условия в начале цикла

дает результат “ложь”. Поэтому тело цикла больше не выполняется, и, поскольку оператор цикла в программе является последним, вычисления заканчиваются.

```
program P1;
      { координата вертикально брошенного тела }
const
  Vo = 5; g = 9.81;
  Tmax = 1; Step = 0.1;
var
  y, t: real;

BEGIN
  t:= 0;
  while t <= Tmax do
    begin
      y:= Vo*t - g*Sqr(t)/2;
      Writeln (t, y);
      t:= t + Step
    end
END.
```

Программа P1 выводит на экран следующую таблицу:

```
 0.0000000000E+00 0.0000000000E+00
 1.0000000000E-01 4.5095000000E-01
 2.0000000000E-01 8.0380000000E-01
 3.0000000000E-01 1.0585500000E+00
 4.0000000000E-01 1.2152000000E+00
 5.0000000000E-01 1.2737500000E+00
 6.0000000000E-01 1.2342000000E+00
 7.0000000000E-01 1.0965500000E+00
 8.0000000000E-01 8.6080000000E-01
 9.0000000000E-01 5.2695000000E-01
```

Мы видим, что результаты работы программы представлены в форме, неудобной для восприятия. Это легко исправить, задав в процедуре `Writeln` нужный формат вывода чисел с помощью двоеточий, как это будет показано далее.

Более существенно то, что в таблице отсутствует последняя строка, соответствующая значению $t = 1$, хотя условие `t <= Tmax` при этом, казалось бы, выполняется. Этот довольно тонкий эффект связан с тем, что компьютер числа типа real иногда записывает приближенно. Например, число 0.1 (одна десятая) в двоичной системе счисления представляет собой бесконечную периодическую дробь

(также, как 1/3 в десятичной системе записывается в виде 0.33333333...). Но в памяти компьютера число всегда округляется до конечного количества значащих цифр. В нашем случае это приводит к тому, что после десятикратного прибавления Step = 0.1 к текущему значению времени получается величина не равная, а немного превышающая единицу. Это вызывает преждевременный выход из цикла. Своевременный выход из цикла гарантируется, если воспользоваться условием вида `t < Tmax + Step/2`. Вообще, для надежной работы программ следует избегать знака равенства при сравнении чисел типа real.

Описанные выше недостатки исправлены в программе P2. Кроме того, в ней организован вывод заголовка таблицы и пустой строки, отделяющей заголовок от самой таблицы. Процедура `Writeln (t:10:1, y:10:3)` выделяет для вывода значений `t` и `y` по 10 знакомест на экране, причем значение `t` выводится с одним знаком после десятичной точки, а значение `y` - с тремя знаками.

В конце программы P2 стоит процедура `Readln`. При ее выполнении компьютер останавливается и ждет нажатия на клавишу Enter. Это дает возможность рассмотреть выведенные на экран результаты, поскольку иначе сразу после завершения программы на экране появляется окно с текстом программы. (Напомним, что результаты всегда можно просмотреть и с помощью клавиш Alt+F5).

```
program P2;
      { координата вертикально брошенного тела }
const
  Vo = 5; g = 9.81;
  Tmax = 1; Step = 0.1;
var
  y, t: real;

BEGIN
  Writeln ('      ВРЕМЯ    ВЫСОТА');
  Writeln;

  t:= 0;
  while t < Tmax + Step/2 do
    begin
      y:= Vo*t - g*Sqr(t)/2;
      Writeln (t:10:1, y:10:3);
      t:= t + Step
    end;
```

```
  Readln
END.
```

Результаты работы программы P2.

```
      ВРЕМЯ    ВЫСОТА

       0.0     0.000
       0.1     0.451
       0.2     0.804
       0.3     1.059
       0.4     1.215
       0.5     1.274
       0.6     1.234
       0.7     1.097
       0.8     0.861
       0.9     0.527
       1.0     0.095
```

2. Использование функций

Предположим, что нам необходимо рассчитать зависимость координаты вертикально брошенного тела от времени не при одном, а при трех значениях начальной скорости, например: v_0 = 2 м/с, v_0 = 5 м/с и v_0 = 10 м/с. Для решения этой задачи, конечно, можно в теле цикла трижды записать формулу (1) с тремя различными значениями начальной скорости. Однако такой способ не является рациональным, особенно если вычисления приходится проводить по более сложным формулам. Язык Pascal дает возможность запрограммировать формулу один раз, и затем подставлять в нее любые значения параметров.

Будем считать, что высота подъема тела, рассчитываемая по формуле (1), представляет собой *функцию* двух переменных: $y(v_0, t)$.

Описания функций в программе должны располагаться сразу после описания переменных. Описание в простейшем случае имеет вид:

```
function <имя>(<аргументы>: тип аргументов): тип функции;
  begin
    <операторы вычисления функции>
  end;
```

Для нашей задачи описание функции *y* будет таким:

```
function y (Vo,t: real): real;
  begin
    y:= Vo*t - g*Sqr(t)/2
  end;
```

Если теперь в программе где-либо потребуется вычислить данную функцию при определенных значениях аргументов, достаточно будет указать имя функции, а затем в скобках значения аргументов в том порядке, в каком они следуют в описании. При этом программа автоматически выполнит операторы, указанные в описании функции, и возвратит вычисленное значение в ту точку программы, откуда функция вызвана. Например, выражение

```
Writeln (y (5, 0.5):10:3);
```

выведет на экран значение высоты тела, брошенного с начальной скоростью 5 м/с, через 0.5 секунды после броска.

Ниже приводится текст программы P3 для расчета высоты подъема тела при трех значениях начальной скорости и результаты ее работы. Значения начальной скорости заданы в виде констант `Vo1`, `Vo2`, `Vo3`. Обратите также внимание на способ вывода заголовка таблицы. Числа после двоеточий указывают количество выделяемых каждому слову знакомест, причем свободные места компьютер заполняет пробелами слева от слова.

```
program P3;
      { координата вертикально брошенного тела }
const
  g = 9.81;
  Vo1 = 2; Vo2 = 5; Vo3 = 10;
  Tmax = 1; Step = 0.1;
var
  t: real;
function y (Vo,t: real): real;
  begin
    y:= Vo*t - g*Sqr(t)/2
  end;

BEGIN
  Writeln ('ВРЕМЯ':11, 'Vo = 2':10, 'Vo = 5':10, 'Vo =
10':10);
  Writeln;

  t:= 0;
```

```
  while t < Tmax + Step/2 do
    begin
      Writeln (t:10:1,
               y(Vo1,t):10:3, y(Vo2,t):10:3,
y(Vo3,t):10:3);
      t:= t + Step
    end;
  Readln
END.
```

ВРЕМЯ	Vo = 2	Vo = 5	Vo = 10
0.0	0.000	0.000	0.000
0.1	0.151	0.451	0.951
0.2	0.204	0.804	1.804
0.3	0.159	1.059	2.559
0.4	0.015	1.215	3.215
0.5	-0.226	1.274	3.774
0.6	-0.566	1.234	4.234
0.7	-1.003	1.097	4.597
0.8	-1.539	0.861	4.861
0.9	-2.173	0.527	5.027
1.0	-2.905	0.095	5.095

3. Несколько полезных примеров

Набор встроенных математических функций языка Turbo-Pascal сравнительно беден: синус, косинус, арктангенс, натуральный логарифм, экспоненциальная функция, квадратный корень и немногие другие. Это сделано вполне сознательно. Гибкость языка позволяет программировать любые сколь угодно сложные функции и в дальнейшем обращаться к ним, будто это встроенные функции. Ниже приводится три фрагмента программ с описаниями функций, часто встречающихся в физике.

Тангенс.
Вычисление основано на использовании тождества

$$\operatorname{tg} x = \frac{\sin x}{\cos x} \quad .$$

```
function tg (x: real): real;
      { функция тангенс }
begin
  tg:= Sin(x)/Cos(x)
end;  { tg }
```

Кубический корень.
Вычисление основано на использовании тождества

$$\sqrt[3]{x} = e^{\frac{\ln x}{3}} \qquad (x > 0) \quad .$$

```
function Qrt (x: real): real;
      { функция кубический корень }
begin
  Qrt:= Exp(Ln(x)/3)
end;  { Qrt }
```

Возведение в дробную степень.
Вычисление основано на использовании тождества

$$x^y = e^{y \ln x} \qquad (x > 0) \quad .$$

```
function Pow (x,y: real): real;
      { степенная функция }
begin
  Pow:= Exp(y*Ln(x))
end;  { Pow }
```

4. Вывод результатов расчетов в текстовый файл

Файл представляет собой определенным образом организованную запись на жестком диске компьютера или на дискете. В виде файлов хранятся как программы, по которым работает компьютер, так и любые данные (числовые данные, тексты, изображения и др.).

Существуют файлы различных типов, которые отличаются способом организации записанной в них информации. Простейший тип - текстовые файлы. Они представляют собой набор символов, то есть букв, цифр и других знаков. Содержимое файла легко просмотреть на экране (например, с помощью Norton Commander, используя клавишу F3) или вывести на печать. В текстовый файл удобно записывать результаты вычислений, особенно если они целиком не помещаются на экране компьютера. К тому же существуют сервисные программы, позволяющие быстро

строить графики на основе табличных данных, записанных в текстовом файле.

Для вывода информации в текстовый файл в программе, написанной на языке Turbo-Pascal, необходимо предусмотреть следующие операции.

1. В разделе описания переменных в начале программы необходимо *объявить файловую переменную*, которая должна иметь тип `text`.

Например:

```
var
  OutFile: text;
```

Здесь `OutFile` - имя файловой переменной (может быть любым).

2. Необходимо *связать файловую переменную с конкретным именем файла* на диске. Для этого служит процедура `Assign`, в которой нужно указать два параметра: используемую файловую переменную и имя файла на диске.

Например, процедура вида

```
Assign (OutFile, 'RESULT.DAT');
```

связывает файловую переменную `OutFile` с дисковым файлом, имя которого `RESULT.DAT` (напомним, что символы после точки называют расширением файла). Важно, чтобы имя файла было взято в апострофы.

После выполнения процедуры `Assign` любые действия в программе, где упоминается файловая переменная `OutFile`, будут происходить с файлом по имени `RESULT.DAT`.

3. Далее необходимо *открыть файл для записи*. Для этого служит процедура `Rewrite`, параметром которой должна быть файловая переменная.

Например, процедура

```
Rewrite (OutFile);
```

создает на диске пустой файл с именем `RESULT.DAT` и создает все условия для записи в него информации.

4. *Запись в файл*. Она осуществляется обычными процедурами `Write` и `Writeln` с единственным отличием:

первым параметром процедуры ставится файловая переменная.

Так, процедура

```
Writeln (X:10, Y:10);
```

выводит численные значения переменных X и Y на экран, а процедура

```
Writeln (OutFile, X:10, Y:10);
```

записывает эти данные в файл (в нашем случае `RESULT.DAT`).

5. После того, как все действия с файлом закончены, то есть в конце программы, его необходимо *закрыть*. При этом в специальную область на диске автоматически заносится служебная информация о файле (его длине, дате и времени создания и пр.). Закрывает файл процедура `Close`, параметром которой является файловая переменная.

Например:

```
Close (OutFile);
```

Ниже в качестве примера приводится программа, создающая текстовый файл с именем LYCEUM.TXT и записывающая в него слова “Ришельевский лицей”.

```
program MAKEFILE;
      {пример создания текстового файла}
var
  OutF: text;

BEGIN
  Assign (OutF, 'LYCEUM.TXT');
  Rewrite (OutF);

  Writeln (OutF, 'Ришельевский лицей');

  Close (OutF)
END.
```

Еще один пример - программа P4. Она полностью аналогична программе P3, но выводит результаты вычислений не на экран, а в текстовый файл FALL.DAT.

```
program P4;
      { координата вертикально брошенного тела }
const
  g = 9.81;
```

```
  Vo1 = 2; Vo2 = 5; Vo3 = 10;
  Tmax = 1; Step = 0.1;
var
  t: real;
  OutFile: text;
function y (Vo,t: real): real;
  begin
    y:= Vo*t - g*Sqr(t)/2
  end;

BEGIN
  Assign (OutFile, 'FALL.DAT');
  Rewrite (OutFile);

  Writeln (OutFile, 'ВРЕМЯ':11,
                    'Vo = 2':10, 'Vo = 5':10, 'Vo =
10':10);
  Writeln (OutFile);

  t:= 0;
  while t < Tmax + Step/2 do
    begin
      Writeln (OutFile, t:10:1,
                  y(Vo1,t):10:3, y(Vo2,t):10:3,
y(Vo3,t):10:3);
      t:= t + Step
    end;
  Close (OutFile)
END.
```

В заключение отметим, что на стадии отладки программы удобнее результаты вычислений выводить на экран, даже если в окончательном варианте они будут выводиться в файл. Для этого проще всего при отладке в процедуре `Assign` вместо имени дискового файла временно указать специальное имя, закрепленное за экраном компьютера: ‘CON’. Например, в программе P4 записать

```
Assign (OutFile, 'CON');
```

В этом случае для перехода к “чистовому” варианту программы после устранения всех ошибок останется заменить ‘CON’ на имя файла на диске.

ВАРИАНТЫ ЗАДАНИЙ

Вариант 1

Сложение взаимно перпендикулярных колебаний

Материальная точка совершает колебания в плоскости (x0y) по закону:

$$x(t) = A \cdot \sin 4\pi t$$
$$y(t) = A \cdot \cos 5\pi t$$

Рассчитайте зависимости $x(t)$ и $y(t)$ для промежутка $0 \le t \le 2$ с шагом изменения t 0,01. Результаты расчета в виде таблицы выведите в текстовый файл с именем LISS.DAT следующего вида:

```
    t          X          Y

 0.00       0.00      50.00
 0.01       6.27      49.38
 0.02      12.43      47.55
 0.03      18.41      44.55
 0.04      24.09      40.45
 0.05      29.39      35.36
 ........................
 1.95     -29.39      35.36
 1.96     -24.09      40.45
 1.97     -18.41      44.55
 1.98     -12.43      47.55
 1.99      -6.27      49.38
 2.00       0.00      50.00
```

Изобразите на бумаге траекторию движения материальной точки (фигуру Лиссажу). Амплитуда A = 50 мм.

Вариант 2

Гармонический синтез прямоугольных колебаний

Периодические колебания произвольной формы можно представить в виде суммы гармонических колебаний (гармоник), частоты которых кратны частоте исходных колебаний. Определение амплитуд и начальных фаз гармоник называется *гармоническим анализом* или разложением в ряд Фурье. Восстановление исходного колебания путем сложения гармоник называется *гармоническим синтезом*. Синтезированные колебания тем меньше отличаются от исходных, чем большее число гармоник используется при синтезе.

Например, колебания прямоугольной формы можно представить рядом Фурье вида

$$F(t) = \frac{2}{\pi} \cdot \left(\sin t + \tfrac{1}{3}\sin 3t + \tfrac{1}{5}\sin 5t + \ldots\right) \quad .$$

Проведите гармонический синтез прямоугольных колебаний с различной степенью точности, используя один, два и три члена ряда Фурье. Полученные зависимости в виде таблицы выведите в текстовый файл с именем FOURIER.DAT по следующему образцу:

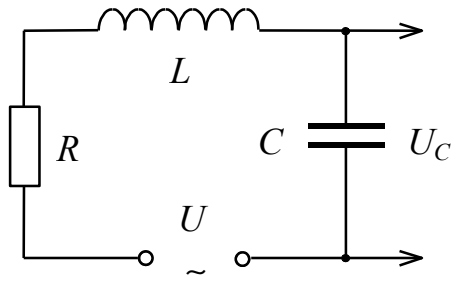
```
   T         F1         F2         F3

 0.00      0.000      0.000      0.000
 0.12      0.076      0.151      0.223
 0.24      0.151      0.291      0.410
 0.36      0.224      0.411      0.535
.....................................
11.40     -0.585     -0.511     -0.455
11.52     -0.551     -0.552     -0.441
11.64     -0.509     -0.584     -0.457
11.76     -0.459     -0.600     -0.501
11.88     -0.403     -0.591     -0.554
12.00     -0.342     -0.552     -0.591
```

При вычислениях время изменяйте в пределах $0 \le t \le 12$ с, шаг 0.12 с. Постройте на бумаге графики синтезированных колебаний. Чему равны их амплитуда и период?

Вариант 3

Резонансные кривые колебательного контура

Ток I в последовательном колебательном контуре, подключенном к источнику переменного напряжения с действующим значением U и частотой ν, определяется законом Ома для цепи переменного тока



$$I = \frac{U}{\sqrt{R^2 + \left(\omega L - \frac{1}{\omega C}\right)^2}}$$

где L, C, R - соответственно, индуктивность, емкость и активное сопротивление контура, $\omega = 2\pi\nu$ - угловая частота изменения напряжения.

Напряжение на конденсаторе контура U_C можно найти, умножив ток в контуре на емкостное сопротивление конденсатора X_C:

$$U_C = I \cdot X_C = I \cdot \frac{1}{\omega C}\ ,\ \text{откуда}\ \ U_C = \frac{U}{\omega C \cdot \sqrt{R^2 + (\omega L - 1/\omega C)^2}}$$

Рассчитайте зависимости $U_C(\nu)$ (резонансные кривые контура) для значений R = 20 Ом, 50 Ом и 100 Ом в диапазоне ν 0.025 - 2.5 МГц (шаг 0.025 МГц) при следующих значениях параметров: C =1.2 нФ, L =14.7 мкГн, U =1В. Таблицу результатов выведите в текстовый файл с именем RESON.DAT следующей структуры:

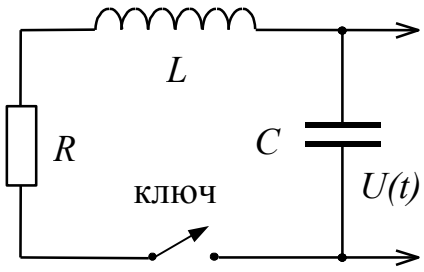
```
   V        R=20       R=50       R=100

0.025      1.000      1.000      1.000
0.050      1.002      1.002      1.001
0.075      1.004      1.004      1.002
0.100      1.007      1.006      1.004
0.125      1.011      1.010      1.006
.....................................
2.400      0.330      0.318      0.285
2.425      0.321      0.310      0.278
2.450      0.312      0.302      0.272
```

```
2.475      0.304      0.294      0.266
2.500      0.296      0.287      0.260
```

По таблице постройте на бумаге резонансные кривые колебательного контура. Рассчитайте и отметьте на графике частоту, при которой максимален ток в контуре. Будет ли при этом максимальным напряжение на конденсаторе?

Вариант 4

Затухающие колебания в колебательном контуре



После замыкания изображенного на схеме ключа конденсатор, предварительно заряженный от внешнего источника до напряжения U_0, начинает разряжаться на резистор сопротивлением R и катушку индуктивности L. При этом напряжение на конденсаторе со временем изменяется по закону

$$U(t) = U_0 \cdot \mathrm{e}^{-\delta t} \cdot \cos \omega t, \text{ где } \quad \delta = \frac{R}{2L}; \ \omega = \sqrt{\frac{1}{LC} - \frac{R^2}{4L^2}} \ .$$

Рассчитайте зависимости $U(t)$ для промежутка времени $0 \le t \le 10$ мс с шагом 0.1 мс в колебательном контуре с L = 40 мГн, C = 4.8 мкФ для трех значений сопротивления R = 10 Ом, 30 Ом, 100 Ом. U_0 = 10 В. Результаты выведите в текстовый файл с именем DAMPING.DAT в виде таблицы следующего образца:

```
Time      R=10       R=30       R=100

 0.0     10.000     10.000     10.000
 0.1      9.620      9.389      8.665
 0.2      8.758      8.353      7.227
 0.3      7.468      6.974      5.776
 0.4      5.827      5.347      4.381
 ..................................
 9.5     -2.872     -0.233      0.000
 9.6     -2.992     -0.254      0.000
 9.7     -2.956     -0.260      0.000
```

9.8	-2.769	-0.253	0.000
9.9	-2.445	-0.234	0.000
10.0	-2.004	-0.204	0.000

На бумаге вычертите график полученных зависимостей.

Вариант 5

Изотермы реального газа

Простейшее уравнение состояния реального газа – уравнение Ван-дер-Ваальса имеет вид:

$$\left(p+\frac{a}{v^2}\right)\cdot(v-b)=RT \ ,$$

где p - давление газа, v - его молярный объем, T - температура,

R - универсальная газовая постоянная.

a и b - так называемые константы Ван-дер-Ваальса, которые могут быть определены по критическим параметрам вещества – критической температуре $T_к$, критическому молярному объему $v_к$ и давлению в критической точке $p_к$ по формулам:

$$a=3p_к v_к^2 \ ; \quad b=\tfrac{1}{3}v_к \ ; \quad R=\tfrac{8}{3}\cdot\frac{v_к p_к}{T_к} \ .$$

Уравнение Ван-дер-Ваальса можно записать в виде, который не зависит от конкретного реального газа, т.е. от констант a и b. Для этого вводят величины

$\pi = p/p_к$ – приведенное давление,

$\varphi = v/v_к$ – приведенный объем,

$\tau = T/T_к$ – приведенная температура.

Можно показать, что в этих обозначениях уравнение состояния реального газа имеет вид:

$$\pi=\frac{8\tau-3(3\varphi-1)/\varphi^2}{3\varphi-1} \ .$$

Это уравнение называется *приведенным уравнением Ван-дер-Ваальса*, а графики зависимости $\pi(\varphi)$ называются для разных τ *приведенными изотермами*.

Рассчитайте приведенные изотермы Ван-дер-Ваальса в области изменения приведенного объема $0.4 \le \varphi \le 2.4$ с шагом 0.05 для трех значений приведенной температуры: 0.85, 1.0 и 1.3. Таблицу полученных значений выведите в текстовый файл с именем ISOTHERM.DAT следующего вида:

```
 Phi     Tau=0.85  Tau=1.0   Tau=1.3

0.40     15.250    21.250    33.250
0.45      4.614     8.042    14.899
0.50      1.600     4.000     8.800
0.55      0.544     2.390     6.083
0.60      0.167     1.667     4.667
0.65      0.057     1.320     3.847
.................................
2.20      0.594     0.809     1.237
2.25      0.590     0.799     1.216
2.30      0.585     0.789     1.196
2.35      0.581     0.779     1.176
2.40      0.576     0.769     1.157
```

По данным из файла постройте на бумаге график приведенных изотерм. Изображайте только те участки изотерм, для которых приведенное давление не превышает значения 4.

Вариант 6

Распределение молекул газа по скоростям

Предположим, в закрытом сосуде имеется n молекул газа. Молекулы имеют различные скорости. Обозначим через Δn число молекул, скорости которых лежат в небольшом интервале от v до $v+\Delta v$. Очевидно, Δn пропорционально величинам n и Δv. Величина коэффициента пропорциональности должна зависеть от того, вблизи какого значения v выбран интервал Δv. Больше всего молекул имеют скорости, не слишком отличающиеся от средней, а доля очень быстрых и очень медленных молекул сравнительно невелика. Таким образом, коэффициент пропорциональности является функцией скорости $f(v)$. Она

называется функцией распределения молекул газа по скоростям.

$$f(v) = \frac{\Delta n}{n \cdot \Delta v} \ .$$

Фактически это доля молекул со скоростями от v до v+1.

Максвелл показал, что

$$f(v) = 4\pi \cdot \left(\frac{M}{2\pi RT} \right)^{\frac{3}{2}} \cdot v^2 \cdot \mathrm{e}^{-\frac{Mv^2}{2RT}} \ .$$

Здесь M - молярная масса газа, T - абсолютная температура, R = 8.31 Дж/(моль·К) - универсальная газовая постоянная. Приведенная зависимость носит название *распределение Максвелла*.

Рассчитайте распределение Максвелла при комнатной температуре (T = 300 К) для молекул неона Ne (M = 0.020 кг/моль) и молекул углекислого газа CO_2 (M = 0.044 кг/моль). Скорость изменяйте в интервале $0 \le v \le 1000$ м/с с шагом 20 м/с. Результаты выведите в текстовый файл с именем MAXWELL.DAT в виде таблицы следующего образца:

```
V, m/s       F(Ne)          F(CO2)

   0      0.000E+00      0.000E+00
  20      7.240E-06      2.358E-05
  40      2.882E-05      9.333E-05
  60      6.433E-05      2.063E-04
  80      1.131E-04      3.578E-04
 100      1.742E-04      5.416E-04
 ..............................
 900      5.699E-04      3.769E-05
 920      5.146E-04      2.856E-05
 940      4.628E-04      2.147E-05
 960      4.144E-04      1.601E-05
 980      3.696E-04      1.185E-05
1000      3.284E-04      8.700E-06
```

На бумаге постройте график распределения Максвелла и найдите наиболее вероятную скорость молекул неона и углекислого газа при указанной температуре.

Вариант 7

Излучение абсолютно черного тела

Нагретые тела излучают энергию (светятся). Обозначим энергию, излучаемую с 1 м^2 поверхности тела за время 1 с через W. Эта энергия излучается в виде волн всевозможной длины (спектр излучения называют сплошным). Обозначим через ΔW энергию, приходящуюся на небольшой интервал длин волн от λ до $\lambda+\Delta\lambda$. Очевидно, ΔW прямо пропорциональна $\Delta\lambda$, а величина коэффициента пропорциональности (обозначим его r) зависит от значения λ, вблизи которого выбран интервал. Например, докрасна нагретое тело большую часть энергии излучает в виде инфракрасных и красных лучей, и очень небольшую – в виде синих и фиолетовых. Кроме того, величина r зависит от температуры тела T, то есть представляет собой функцию двух переменных $r(\lambda,T)$. Она называется *спектральной плотностью излучения*.

$$r(\lambda,T)=\frac{\Delta W}{\Delta\lambda}\ .$$

Фактически это мощность, излучаемая с единицы поверхности тела в единичном спектральном интервале.

Если тело полностью поглощает падающее на него излучение, его называют *абсолютно черным телом*. Макс Планк получил следующее выражение для спектральной плотности излучения абсолютно черного тела:

$$r(\lambda,T)=\frac{2\pi c^2 h}{\lambda^5}\cdot\frac{1}{e^{\frac{hc}{kT\lambda}}-1}$$ (формула Планка).

Здесь $c = 3\cdot10^8$ м/с - скорость света, $h = 6.62\cdot10^{-34}$ Дж.с – постоянная Планка, $k = 1.38\cdot10^{-23}$ Дж/К – постоянная Больцмана.

Рассчитайте по формуле Планка зависимость спектральной плотности излучения абсолютно черного тела от длины волны в диапазоне от 50 нм до 1000 нм с шагом 25 нм для трех значений температуры T = 5000 К, 6000 К, 7000 К. Полученные результаты выведите в текстовый файл с именем PLANCK.DAT в виде таблицы следующего вида:

L	T=5000	T=6000	T=7000
50	1.197E-04	1.757E+00	1.664E+03
75	3.397E+03	2.037E+06	1.964E+08
100	1.183E+07	1.434E+09	4.412E+10
125	1.226E+09	5.693E+10	8.827E+11
150	2.288E+10	5.602E+11	5.500E+12
.....			
900	2.699E+13	4.742E+13	7.189E+13
925	2.576E+13	4.469E+13	6.716E+13
950	2.457E+13	4.211E+13	6.278E+13
975	2.341E+13	3.969E+13	5.871E+13
1000	2.231E+13	3.741E+13	5.494E+13

Вычертите на бумаге график зависимостей r (λ) для указанных значений температуры. Из графика определите длины волн, соответствующие максимумам спектральной плотности излучения для каждой температуры.

*Вариант 8**

Броуновское движение

За броуновской частицей ведется наблюдение через микроскоп с увеличением N = 1000 и через равные промежутки времени отмечаются ее координаты x и y. Затем отмеченные точки последовательно соединяются ломаной линией.

Рассчитайте и изобразите на бумаге эту ломаную линию, исходя из того, что за время между отсчетами частица смещается на расстояние S = 10 мкм в случайном направлении. Можно показать, что при этом ее координаты x и y изменяются на величину

$$\Delta x = S \cdot \sin \pi R_1 \cdot \cos 2\pi R_2$$
$$\Delta y = S \cdot \sin \pi R_1 \cdot \sin 2\pi R_2 \quad ,$$

где R_1 и R_2 - два случайных числа, равномерно распределенных на интервале (0,1).

Вычисления прекратите, когда частица выйдет за пределы поля зрения микроскопа - круга радиусом 50 мм с центром в начале координат. Границу поля зрения изобразите пунктирной линией. Начальное положение частицы $x = y = 0$.

Результаты расчетов выведите в текстовый файл с именем BROWN.DAT в виде следующей таблицы[1]:

```
        x          y

     0.00       0.00
     6.77      -7.36
     8.34       2.13
     4.17      -0.77
     2.58       3.03
.................
-40.02      -8.37
-41.85      -8.89
-47.23      -1.32
-47.52       5.97
-47.40       5.78
```

Подсказка

Для нахождения случайных чисел R_1 и R_2 воспользуйтесь стандартной функцией Паскаля `Random`.

* * *

[1] Результаты расчетов наверняка не совпадут с приведенными в таблице данными, поскольку Вы моделируете случайный процесс.

Глава 2

Численное решение основной задачи механики

1. *Как компьютер решает уравнения Ньютона?*

Решить основную задачу механики, – значит, уметь указать в любой момент времени координату и скорость тела. Для этого необходимо знать:

- координату x_0 и скорость v_0 в начальный момент времени t_0 (*начальные условия*);
- зависимость силы, действующей на тело, (точнее, равнодействующей всех приложенных сил) от его координаты, скорости и, возможно, от времени $F(x, v, t)$.

В простейшем случае равнодействующая приложенных к телу сил равна нулю. Тогда в соответствии со вторым законом Ньютона ускорение тела равно нулю, и движение тела будет равномерным. Задача становится немного сложнее, если на тело действует постоянная сила, и поэтому ускорение не равно нулю, но остается неизменным во времени. Иными словами движение тела будет равноускоренным. В обоих случаях для нахождения координаты и скорости тела существуют известные формулы, в которые достаточно подставить начальные условия.

Но предположим, что сила, действующая на тело, изменяется в ходе его движения, и, следовательно, ускорение тела не является постоянным. Тогда задача серьезно усложняется и далеко не всегда имеет аналитическое решение. В таких случаях используют различные численные методы, для реализации которых

обычно используется компьютер, поскольку эти методы требуют выполнения большого числа однотипных вычислений. Рассмотрим простейшие из них.

Выберем промежуток времени Δt настолько малый, что за это время сила, действующая на тело, а значит, и его ускорение, практически не изменяются. Тогда по известной формуле для равноускоренного движения скорость тела в следующий момент времени $t_0+\Delta t$ равна

$$v_1 = v_0 + a(x_0, v_0, t_0) \cdot \Delta t \quad ,$$

или, принимая во внимание второй закон Ньютона[1],

$$v_1 = v_0 + \frac{F(x_0, v_0, t_0)}{m} \cdot \Delta t \qquad (1).$$

Теперь, когда известна скорость тела в новый момент времени, определим координату тела в этот момент:

$$x_1 = x_0 + v \cdot \Delta t \qquad (2).$$

Какое значение скорости – начальное v_0 или новое v_1 подставить в качестве v в формуле (2), не имеет существенного значения, если величина Δt достаточно мала. Наиболее простыми являются такие методы определения скорости и координаты:

$$\begin{cases} v_1 = v_0 + \dfrac{F(x_0, v_0, t_0)}{m} \cdot \Delta t \\ x_1 = x_0 + v_0 \cdot \Delta t \end{cases} \quad (3) \ – \textit{метод Эйлера},$$

$$\begin{cases} v_1 = v_0 + \dfrac{F(x_0, v_0, t_0)}{m} \cdot \Delta t \\ x_1 = x_0 + v_1 \cdot \Delta t \end{cases} \quad (4) \ – \textit{метод Эйлера-Кромера}.$$

Детальный анализ показывает, что метод Эйлера-Кромера дает несколько большую точность результатов при одинаковом шаге Δt.

Получив значения координаты и скорости в момент $t_0+\Delta t$, можно использовать их как начальные для расчета координаты и скорости в следующий момент $t_0+2\Delta t$ и т.д.

[1] Масса тела предполагается неизменной.

Таким образом, *многократное* применение формул (3) или (4) дает возможность найти значения $v(t)$ и $x(t)$ в *любой* момент времени, то есть решить основную задачу механики. Это решение является приближенным, но погрешность решения можно сделать сколь угодно малой, если выбрать достаточно малые значения Δt.

Ниже приведен фрагмент программы на языке Pascal, реализующий метод Эйлера-Кромера и вычисляющий координату и скорость тела в момент времени, отстоящий на 1 секунду от начального:

```
dt:= 0.001;
for i:= 1 to 1000 do
  begin
    V:= V + F(X,V,t)/m*dt;
    X:= X + V*dt;
    t:= t + dt
  end;
Writeln (t:10:0, V:10:2, X:10:2);
```

Здесь промежуток времени 1 секунда разбит на 1000 элементарных участков Δt (в программе обозначенных `dt`), для каждого из которых использован метод Эйлера-Кромера. В каждом цикле новые значения координаты и скорости записываются на место старых значений, т.е. используются в качестве начальных для последующего цикла.

Понятно, что для работы этого фрагмента необходимо:

– предварительно описать все входящие во фрагмент переменные и константу `m`,
– описать функцию `F(X,V,t)`, которая вычисляет равнодействующую приложенных к телу сил,
– присвоить начальные значения времени, координаты и скорости.

Данный фрагмент можно разместить внутри другого цикла, и тогда получится программа, выводящая таблицу значений координаты и скорости тела каждую секунду. По ней можно построить графики зависимостей $v(t)$ и $x(t)$ и, таким образом, судить о характере движения тела.

Описанный метод подходит для решения любых задач об одномерном движении тела, даже тех, которые не имеют аналитического решения (например, задача о колебаниях маятника с большой амплитудой). При этом *одна задача от другой отличается только видом зависимости*

равнодействующей $F(x,v,t)$, которая устанавливается из анализа всех сил, действующих на тело.

Один из важных вопросов, требующих разрешения: насколько малым должен быть шаг численного метода Δt, чтобы обеспечить нужную точность вычислений? Рассмотрим два способа выбора Δt.

Метод пробного шага.

Вычисления проводят дважды со значениями Δt, отличающимися в несколько (обычно в 2 или 10) раз и результаты сравнивают. Если результаты совпадают в требуемом числе значащих цифр, то больший шаг обеспечивает необходимую точность (меньший и подавно). В противном случае шаг уменьшают и повторяют пробу.

Контроль сохранения полной энергии.

Если рассматриваемая система замкнута и в ней не действуют силы трения, то ее полная механическая энергия должна оставаться неизменной. В ходе расчетов постоянно вычисляют сумму потенциальной и кинетической энергии и выводят наряду со значениями координат и скоростей. Если полная энергия остается неизменной с точностью до нужного числа значащих цифр, то с такой же точностью получены значения координат и скоростей. В противном случае требуется уменьшить шаг.

2. Пример построения законченной программы

В качестве примера рассмотрим все этапы написания программы, рассчитывающей движение пружинного маятника методом Эйлера-Кромера.

Пусть масса груза маятника $m = 100$ г, жесткость пружины $k = 2.5$ Н/м. Начальное отклонение груза $x_0 = 0.1$ м, начальная скорость равна нулю. Рассчитаем зависимость от времени скорости груза $v(t)$ и его отклонения от положения равновесия $x(t)$ в интервале $0 \le t \le 2$ с, шаг вывода результатов 0.1 с. Шаг метода Эйлера-Кромера примем равным $\Delta t = 0.001$ с.

Прежде всего, напишем раздел описаний моделирующей программы. Масса груза, жесткость пружины, начальное отклонение, время окончания вычислений, шаг численного

метода являются параметрами, неизменными в ходе движения маятника, т.е. константами. Соответствующий фрагмент программы может иметь вид:

```
const
  m = 0.1; k = 2.5; X0 = 0.1;
  t_end = 2.0; dt = 0.001; Ncalc = 100;
```

В данном фрагменте вместо шага вывода результатов задана величина `Ncalc`. Она подбирается с таким расчетом, чтобы за `Ncalc` шагов протяженностью `dt` пройти интервал, равный одному шагу вывода.

В разделе описания переменных указываем, какие величины характеризуют состояние маятника и изменяются в ходе его движения. В нашем случае это время, скорость и координата:

```
var
  X, V, t: real;
```

Далее задаем в виде функции равнодействующую приложенных к телу сил. У нас на груз действует единственная сила – сила упругости. По закону Гука $F = -kx$, поэтому записываем:

```
function F(X: real): real;
begin
  F:= -k*X
end;
```

В рассматриваемом простейшем случае сила зависит только от координаты. Если бы требовалось учесть сопротивление вязкой среды, добавилась бы зависимость от скорости. В случае вынужденных колебаний появилась бы и зависимость силы от времени.

Наконец, важнейшая часть программы, в которой реализуется алгоритм Эйлера-Кромера. Здесь удобнее всего организовать вычисления с помощью *процедуры*.

Процедура – это "государство в государстве" – относительно самостоятельный участок программы со своим именем, который располагается в разделе описаний. Описание процедуры начинается со слова `procedure`, после чего следует имя процедуры (в нашем случае `Euler`). Мы используем простейший вид процедуры – процедуру без параметров, поэтому вслед за именем сразу ставится точка с запятой. Этим завершается *заголовок* процедуры. Затем следует *тело* процедуры. По своей структуре оно точно такое

же, как сама Pascal-программа: сначала следует (необязательный) раздел описаний, затем раздел операторов, расположенный между словами `begin` и `end`. В отличие от переменных, указанных в разделе описаний программы (они называются *глобальными* переменными), те переменные, которые описаны в разделе описаний процедуры (*локальные* переменные), используются только внутри процедуры. После выполнения процедуры их значения теряются.

```
procedure Euler;
var
  i: integer;
begin
  for i:= 1 to NCalc do
    begin
      V:= V + F(X)/m * dt;
      X:= X + V * dt;
      t:= t + dt
    end
end;
```

Процедура `Euler` проводит вычисления по формулам (4) `Ncalc` раз. Поэтому, как только в основной программе встречается имя `Euler` (говорят, как только эта процедура *вызывается* из основной программы), значение времени увеличивается на величину шага вывода результатов, а значения координаты и скорости соответствуют этому новому значению времени. Таким образом, в разделе операторов основной программы достаточно сначала задать начальные значения переменных, а затем организовать цикл, внутри которого вслед за вызовом процедуры `Euler` будет следовать процедура вывода результатов `Writeln`.

Ниже программа приводится полностью, а также дана распечатка результатов ее работы.

```
program SPRING;

const
  m = 0.1; k = 2.5; X0 = 0.1;
  t_end = 2.0; dt = 0.001; Ncalc = 100;
var
  X, V, t: real;

function F(X: real): real;
begin
  F:= - k*X
end;  { F }
```

```
procedure Euler;
var
  i: integer;
begin
  for i:= 1 to NCalc do
    begin
      V:= V + F(X)/m * dt;
      X:= X + V * dt;
      t:= t + dt
    end
end;  { Euler }

BEGIN
     { задание начальных условий }
  t:= 0;
  X:= X0;
  V:= 0;
     { заголовок таблицы }
  Writeln ('t':9, 'V(t)':11, 'X(t)':10, 'Energy':11);
  Writeln;
     { вывод начальных значений }
  Writeln (t:10:2, V:10:3, X:10:3,
                   k*Sqr(X)/2+m*Sqr(V)/2:11:5);
     { главный цикл }
  repeat
    Euler;
    Writeln (t:10:2, V:10:3, X:10:3,
                     k*Sqr(X)/2+m*Sqr(V)/2:11:5)
  until t > t_end - dt
END.
```

t	V(t)	X(t)	Energy
0.00	0.000	0.100	0.01250
0.10	-0.240	0.088	0.01247
0.20	-0.421	0.054	0.01247
0.30	-0.499	0.007	0.01250
0.40	-0.455	-0.042	0.01252
0.50	-0.299	-0.080	0.01253
0.60	-0.071	-0.099	0.01251
0.70	0.175	-0.094	0.01248
0.80	0.378	-0.065	0.01247
0.90	0.489	-0.021	0.01249
1.00	0.479	0.029	0.01252
1.10	0.353	0.071	0.01253
1.20	0.140	0.096	0.01252
1.30	-0.108	0.098	0.01249
1.40	-0.328	0.075	0.01247
1.50	-0.469	0.034	0.01248
1.60	-0.495	-0.015	0.01251
1.70	-0.399	-0.060	0.01253
1.80	-0.206	-0.091	0.01252
1.90	0.038	-0.100	0.01250

```
2.00      0.272     -0.084     0.01247
```

Для контроля точности в последнем столбце таблицы выводятся значения полной энергии пружинного маятника $E = mv^2/2 + kx^2/2$. Нетрудно видеть, что энергия сохраняется с точностью до трех знаков (с учетом округления).

3. Случай вращательного движения

Для применения численных методов при моделировании вращательного движения воспользуемся известной аналогией между величинами и соотношениями, описывающими поступательное и вращательное движение.

Поступательное движение	*Вращательное движение*
Координата x	Угол φ
Скорость $v = \Delta x/\Delta t$	Угловая скорость $\omega = \Delta\varphi/\Delta t$
Ускорение $a = \Delta v/\Delta t$	Угловое ускорение $\varepsilon = \Delta\omega/\Delta t$
Масса m	Момент инерции I
Сила F	Момент силы M
Второй закон Ньютона $a = F/m$	Второй закон Ньютона $\varepsilon = M/I$

Тогда рассмотренный выше метод Эйлера-Кромера по аналогии со случаем поступательного движения выражается формулами:

$$\begin{cases} \omega_1 = \omega_0 + \varepsilon(\varphi_0, \omega_0, t_0)\cdot\Delta t \\ \varphi_1 = \varphi_0 + \omega_1\cdot\Delta t \end{cases},$$

или, учитывая второй закон Ньютона,

$$\begin{cases} \omega_1 = \omega_0 + \dfrac{M(\varphi_0, \omega_0, t_0)}{I}\cdot\Delta t \\ \varphi_1 = \varphi_0 + \omega_1\cdot\Delta t \end{cases} \qquad (5),$$

где ω_0 и φ_0 – соответственно угловая скорость и угол в начальный момент времени t_0, а ω_1 и φ_1 – угловая скорость и угол в последующий момент $t_0 + \Delta t$.

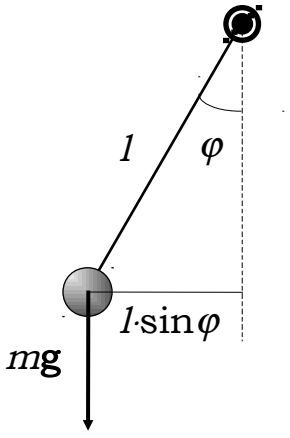
Ниже приведен фрагмент программы на языке Pascal, реализующий метод Эйлера-Кромера и вычисляющий угловое положение и угловую скорость тела в момент времени, отстоящий на 1 секунду от начального:

```
dt:= 0.001;
for ii:= 1 to 1000 do
  begin
    Omega:= Omega + M(Phi,Omega,t)/I*dt;
    Phi:= Phi + Omega*dt;
    t:= t + dt
  end;
Writeln (t:10:0, Omega:10:2, Phi:10:2);
```

Для работы этого фрагмента необходимо:

- предварительно описать все входящие во фрагмент переменные и константу `I`,
- описать функцию `M(Phi,Omega,t)`, которая вычисляет результирующий момент сил, действующих на тело,
- присвоить начальные значения времени, углу и угловой скорости.

В качестве примера рассмотрим задачу о колебаниях маятника на жестком подвесе (отличие от маятника на нити в том, что возможны отклонения от положения равновесия на угол, больший прямого).



Пусть груз массой m = 0.1 кг на жестком невесомом подвесе длиной l = 0.5 м совершает свободные колебания в поле силы тяжести. Ускорение свободного падения g = 9.81 м/с^2.

Получим зависимости угла отклонения от положения равновесия и угловой скорости движения маятника от времени. Начальное отклонение маятника 2,5 рад, начальная угловая скорость нулевая. Расчеты выполним для интервала от 0 до 3 секунд с шагом вывода 0.2 с. Шаг метода Эйлера-Кромера 0.001 с.

АНАЛИЗ ЗАДАЧИ

Угловое ускорение маятника обусловлено действием момента силы тяжести mg. Как видно из рисунка,

$$M = -mgl \cdot \sin\varphi \ .$$

Момент инерции маятника

$$I = ml^2 \ .$$

Поскольку трение в системе отсутствует, то для контроля точности вычислений будем следить за значением полной энергии маятника:

$$E = mgl \cdot (1 - \cos\varphi) + \frac{I\omega^2}{2} \ ,$$

(здесь за нуль принята потенциальная энергия маятника в положении равновесия). Кстати, в тех случаях, когда в системе есть силы трения, их можно временно "отключать" на стадии отладки программы для подбора шага численного метода.

Ниже приводится полный текст моделирующей программы и результаты вычислений.

Работа программы ясна из сравнения с программой для моделирования пружинного маятника. Единственной особенностью является то, что формула для вычисления момента инерции описана в разделе констант. Начиная с версии 6.0 Turbo-Pascal, при описании констант допускается использование арифметических выражений[2].

```
program PENDULUM;
      { моделирование маятника }
const
  mass = 0.1; L = 0.5; g = 9.81;
  Phi0 = 2.5; { начальное отклонение }
  dt = 0.001; Ncalc = 200; t_end = 3.0;
  I = mass*L*L; { момент инерции }

var
  Phi, Omega, t: real;

function M(Phi: real): real;
      { вычисление момента силы }
begin
  M:= -mass*g*L*Sin(Phi)
end;

function E(Phi,Omega: real): real;
```

[2] Но не функций! Поэтому при вычислении квадрата длины в описании констант следует писать `L*L`, а не `Sqr(L)`.

```
      { вычисление полной энергии }
begin
  E:= mass*g*L*(1-Cos(Phi)) + I*Sqr(Omega)/2
end;

procedure Euler;
      { метод Эйлера-Кромера }
var
  ii: integer;
begin
  for ii:= 1 to NCalc do
    begin
      Omega:= Omega + M(Phi)/I * dt;
      Phi:= Phi + Omega * dt;
      t:= t + dt
    end
end;

BEGIN
      { задание начальных условий }
  t:= 0;
  Phi:= 2.5;
  Omega:= 0;
      { заголовок таблицы }
  Writeln ('t':9, 'Omega':11, 'Phi':9, 'Energy':11);
  Writeln;
      { вывод начальных значений }
  Writeln (t:10:1, Omega:10:3,Phi:10:3,
                                E(Phi,Omega):10:3);
      { главный цикл }
  repeat
      Euler;
      Writeln (t:10:1, Omega:10:3, Phi:10:3,
                                     E(Phi,Omega):10:3);
  until t > t_end - dt
END.
```

t	Omega	Phi	Energy
0.0	0.000	2.500	0.883
0.2	-2.587	2.252	0.883
0.4	-6.206	1.386	0.882
0.6	-8.382	-0.150	0.884
0.8	-5.528	-1.596	0.885
1.0	-2.077	-2.335	0.884
1.2	0.409	-2.493	0.883
1.4	3.136	-2.152	0.883
1.6	6.857	-1.159	0.882
1.8	8.188	0.438	0.884
2.0	4.853	1.776	0.885
2.2	1.601	2.398	0.884
2.4	-0.825	2.471	0.883

2.6	-3.724	2.033	0.883
2.8	-7.443	0.910	0.882
3.0	-7.819	-0.716	0.885

ВАРИАНТЫ ЗАДАНИЙ

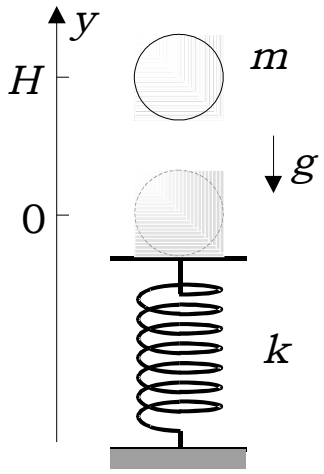
Каждый из вариантов задания состоит из двух пунктов: 1 и 2. Если Вы не очень уверенно программируете, напишите отдельные программы для каждого из пунктов. Для выполнения пунктов 2a, 2b и 2c используйте одну и ту же программу, трижды запуская ее с различными значениями параметров, которые поэтому удобнее всего задавать в разделе констант в самом начале программы. Не забывайте при этом всякий раз изменять имя файла, в который записываются результаты расчетов.

Все задания посвящены исследованию поведения какой-либо колебательной системы (осциллятора). После выполнения расчетов обдумайте ответы на следующие вопросы.

1. Являются ли изучаемые Вами колебания гармоническими? Если нет, то при каких условиях они наиболее близки к ним?
2. Как по виду зависимости потенциальной энергии колеблющегося тела или приложенной к нему силы от координаты предсказать, насколько близкими будут его колебания к гармоническим?
3. Зависит ли период изучаемых Вами колебаний от их амплитуды $T(A)$? Если зависит, то каков качественный характер этой зависимости?
4. Можете ли Вы предсказать, каким будет характер зависимости $T(A)$, по виду зависимости потенциальной энергии колеблющегося тела или приложенной к нему силы от координаты?
5. В чем отличие полученного Вами фазового портрета от фазового портрета гармонических колебаний? Чем объясняется это отличие?
6. Как по виду фазового портрета можно судить о том,
   - является ли движение тела колебательным?
   - действуют ли в системе силы сопротивления?

– каков характер зависимости действующей на тело силы (момента силы) от его координаты?

Вариант 1



Шарик массой $m = 100$ г падает из состояния покоя с высоты H на невесомую горизонтальную площадку, поддерживаемую пружиной с жесткостью $k = 20$ Н/м. Ускорение свободного падения $g = 9.81$ м/с^2.

1. Рассчитайте зависимость равнодействующей сил, приложенных к шарику, F и потенциальной энергии системы E_p от координаты центра шарика в пределах от $y = -0.5$ м до $y = +0.5$ м с шагом 0.02 м. За начало координат примите положение центра шарика, когда он касается недеформированной пружины. Результаты выведите в текстовый файл с именем V1_F&E.DAT в виде таблицы следующего вида:

```
Y,cm     F(Y),N     Ep(Y),J

-50       9.019      2.0095
-48       8.619      1.8331
-46       8.219      1.6647
-44       7.819      1.5044
.........................
 44      -0.981      0.4316
 46      -0.981      0.4513
 48      -0.981      0.4709
 50      -0.981      0.4905
```

2. Методом Эйлера-Кромера рассчитайте зависимости ускорения, скорости, координаты шарика, а также его полной механической энергии от времени для следующих трех случаев:

a. $H = 0$;
b. $H = 0.3$ м;
c. $H = 0.3$ м, причем на шарик дополнительно действует сила сопротивления, пропорциональная его скорости, $F_c = -bv$, где $b = 0.1$ Н·с/м.

Расчеты выполните для интервала $0 \le t \le 5$ с, шаг вывода результатов 0.02 с, шаг метода Эйлера-Кромера $\Delta t = 0.0002$ с. Результаты выведите в текстовые файлы с именами V1_A.DAT, V1_B.DAT, V1_C.DAT в виде таблиц следующего вида:

```
t,s     a,m/s2     V,m/s      Y,m       E,J

0.00    -9.810     0.000     0.300    0.2943
0.02    -9.616    -0.194     0.298    0.2943
0.04    -9.425    -0.385     0.292    0.2941
0.06    -9.239    -0.571     0.283    0.2936
0.08    -9.056    -0.754     0.269    0.2927
0.10    -8.876    -0.934     0.252    0.2912
.........................................
4.90    -0.850     0.250    -0.046   -0.0208
4.92    -1.777     0.224    -0.041   -0.0210
4.94    -2.546     0.180    -0.037   -0.0210
4.96    -3.099     0.123    -0.034   -0.0211
4.98    -3.397     0.058    -0.032   -0.0211
5.00    -3.423    -0.011    -0.032   -0.0211
```

(здесь приведен фрагмент файла V1_C.DAT).

Пользуясь программой EASYPLOT, постройте графики:
- зависимостей $F(y)$ и $E_p(y)$ (два графика на экране);
- зависимостей $y(t)$ - график колебаний, $a(t)$ - график ускорения и $v(y)$ - фазовый портрет (по три графика на экране для каждого из случаев "a", "b", "c").

Подсказка

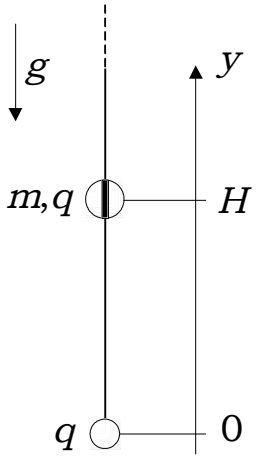
1. Особенность задачи в том, что сила и потенциальная энергия вычисляются по разным формулам для положительных и отрицательных значений y (в зависимости от того, деформирована ли пружина):

$$F(y,v) = \begin{cases} -mg - ky - bv & (y < 0) \\ -mg - bv & (y \ge 0) \end{cases},$$

$$E_p(y) = \begin{cases} mgy + ky^2/2 & (y < 0) \\ mgy & (y \ge 0) \end{cases}.$$

Для программирования этих функций используйте конструкцию `if-then-else`.

Вариант 2



Небольшая бусинка, несущая электрический заряд $q = 20$ нКл, закреплена на нижнем конце расположенной вертикально гладкой неподвижной диэлектрической нити. Вторая бусинка массой $m = 100$ мг с таким же зарядом может свободно скользить по нити.

1. Рассчитайте зависимость равнодействующей F сил, приложенных к верхней бусинке, и потенциальной энергии системы E_p от координаты центра верхней бусинки y в пределах от $y = 2$ см до $y = 15$ см с шагом 2 мм. За начало координат примите положение центра нижней бусинки. Ускорение свободного падения $g = 9.81$ м/с^2. Результаты выведите в текстовый файл с именем V2_F&E.DAT в виде таблицы следующего вида:

```
 Y,cm     F(Y),mN    Ep(Y),mJ

 2.0       8.019      0.1996
 2.2       6.457      0.1852
 2.4       5.269      0.1735
 2.6       4.344      0.1640
 2.8       3.611      0.1560
 3.0       3.019      0.1494
.........................
14.0      -0.797      0.1631
14.2      -0.802      0.1647
14.4      -0.807      0.1663
14.6      -0.812      0.1679
14.8      -0.817      0.1695
15.0      -0.821      0.1711
```

2. Верхнюю бусинку отпускают с высоты H с нулевой начальной скоростью. Методом Эйлера-Кромера рассчитайте зависимости ее ускорения, скорости, координаты, а также

полной механической энергии от времени для следующих трех случаев:

a. $H = 6.5$ см;
b. $H = 20$ см;
c. $H = 20$ см, причем на бусинку дополнительно действует сила сопротивления, пропорциональная ее скорости, $F_c = -bv$, где $b = 1 \cdot 10^{-4}$ Н·с/м.

Расчеты выполните для интервала $0 \le t \le 3$ с, шаг вывода результатов 0.01 с, шаг метода Эйлера-Кромера $\Delta t = 0.0002$ с. Результаты выведите в текстовые файлы с именами V2_A.DAT, V2_B.DAT, V2_C.DAT в виде таблиц следующего вида:

```
t,s      a,m/s2      V,m/s       Y,m       E,mJ

0.00     -8.910     0.0000     0.2000     0.2142
0.01     -8.817    -0.0887     0.1995     0.2142
0.02     -8.717    -0.1763     0.1982     0.2142
0.03     -8.610    -0.2630     0.1960     0.2141
0.04     -8.494    -0.3485     0.1929     0.2140
0.05     -8.370    -0.4329     0.1890     0.2138
.........................................
2.95     -0.454     0.3352     0.0609     0.1245
2.96     -1.418     0.3258     0.0643     0.1244
2.97     -2.200     0.3077     0.0674     0.1243
2.98     -2.826     0.2825     0.0704     0.1242
2.99     -3.317     0.2517     0.0731     0.1241
3.00     -3.694     0.2166     0.0754     0.1241
```

(здесь приведен фрагмент файла V2_C.DAT).

Пользуясь программой EASYPLOT, постройте графики:
– зависимостей $F(y)$ и $E_p(y)$ (два графика на экране);
– зависимостей $y(t)$ - график колебаний, $a(t)$ - график ускорения и $v(y)$ - фазовый портрет (по три графика на экране для каждого из случаев "a", "b", "c").

Подсказки

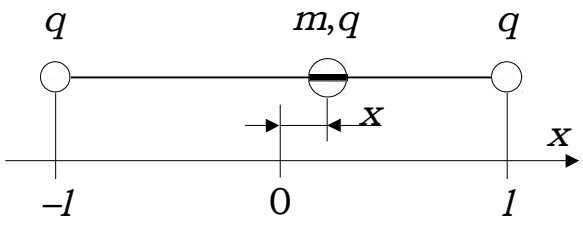
1. На верхнюю бусинку действуют сила тяжести, сила электростатического взаимодействия с нижней бусинкой и, возможно, сила вязкого трения. Поэтому равнодействующая находится по формуле:

$$F(y,v) = -mg + k\frac{q^2}{y^2} - bv \text{ , где } k = 9 \cdot 10^9 \text{ Н·м}^2/\text{Кл}^2 .$$

2. Потенциальная энергия верхней бусинки складывается из энергии гравитационного взаимодействия с Землей и энергии электростатического взаимодействия с нижней бусинкой. Поэтому

$$E_p(y) = mgy + k\frac{q^2}{y} \quad .$$

Вариант 3



Гладкая диэлектрическая нить натянута параллельно горизонтальной оси OX. В точках с координатами $-l$ и $l = 10$ см на нити закреплены точечные электрические заряды величиной $q = 20$ нКл. Между зарядами по нити свободно скользит небольшая бусинка массой $m = 100$ мг, также несущая заряд q.

1. Рассчитайте зависимость равнодействующей F сил, приложенных к бусинке, и потенциальной энергии системы E_p от координаты бусинки x в пределах от $x = -9$ см до $x = 9$ см с шагом 2 мм. За начало координат примите середину участка нити между зарядами. Результаты выведите в текстовый файл V3_F&E.DAT в виде таблицы:

```
X,cm     F(X),mN     Ep(X),mJ

-9.0      35.900      0.3789
-8.8      24.898      0.3191
-8.6      18.263      0.2765
-8.4      13.956      0.2446
.........................
 8.4     -13.956      0.2446
 8.6     -18.263      0.2765
 8.8     -24.898      0.3191
 9.0     -35.900      0.3789
```

2. Бусинка, отпущенная без толчка из точки с координатой x_0, совершает колебания между зарядами. Методом Эйлера-Кромера рассчитайте зависимости ее ускорения, скорости, координаты, а также полной

механической энергии от времени для следующих трех случаев:

a. $x_0 = 2.5$ см;
b. $x_0 = 8.5$ см;
c. $x_0 = 8.5$ см, причем на бусинку дополнительно действует сила сопротивления, пропорциональная ее скорости, $F_c = -bv$, где $b = 2 \cdot 10^{-4}$ Н·с/м.

Расчеты выполните для интервала $0 \le t \le 1.5$ с, шаг вывода результатов 0.005 с, шаг метода Эйлера-Кромера $\Delta t = 0.0002$ с. Результаты выведите в текстовые файлы V3_A.DAT, V3_B.DAT, V3_C.DAT в виде таблиц вида:

t,s	a,m/s2	V,m/s	X,m	E,mJ
0.000	-158.948	0.0000	0.0850	0.2595
0.005	-122.426	-0.7295	0.0830	0.2584
0.010	-70.935	-1.2117	0.0780	0.2574
0.015	-39.176	-1.4821	0.0712	0.2558
0.020	-22.188	-1.6329	0.0634	0.2536
0.025	-12.795	-1.7191	0.0549	0.2509
.....				
1.475	-4.654	-0.0274	0.0278	0.0781
1.480	-4.563	-0.0504	0.0276	0.0781
1.485	-4.449	-0.0730	0.0273	0.0781
1.490	-4.312	-0.0949	0.0269	0.0781
1.495	-4.154	-0.1161	0.0264	0.0781
1.500	-3.977	-0.1364	0.0257	0.0780

(здесь приведен фрагмент файла V3_C.DAT).

Пользуясь программой EASYPLOT, постройте графики:
– зависимостей $F(x)$ и $E_p(x)$ (два графика на экране);
– зависимостей $x(t)$ - график колебаний, $a(t)$ - график ускорения и $v(x)$ - фазовый портрет (по три графика на экране для каждого из случаев "a", "b", "c").

Подсказки

1. Бусинка взаимодействует с левым и правым зарядами. Считая заряды точечными, по закону Кулона запишем формулу для равнодействующей:

$$F(x,v) = k\frac{q^2}{(1+x)^2} - k\frac{q^2}{(1-x)^2} - bv \ ,$$ где $k = 9 \cdot 10^9$ Н·м^2/Кл^2 .

После упрощения имеем:

$$F(x,v) = -4lkq^2 \cdot \frac{x}{\left(l^2 - x^2\right)^2} - bv \quad .$$

2. Потенциальная энергия взаимодействия бусинки с обоими зарядами

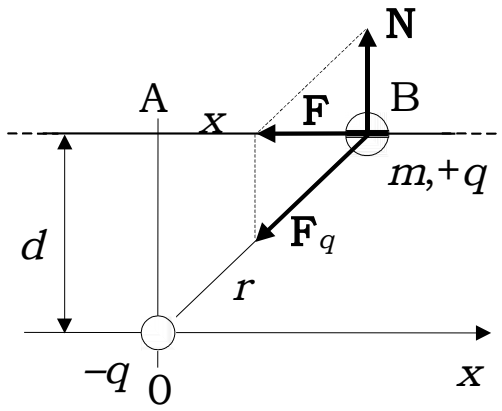
$$E_p(x) = k\frac{q^2}{l+x} + k\frac{q^2}{l-x} \quad .$$

После упрощения:

$$E_p(x) = \frac{2kq^2 l}{l^2 - x^2} \quad .$$

Вариант 4

По гладкой горизонтальной диэлектрической нити скользит небольшая бусинка массой $m = 100$ мг, несущая электрический заряд $q = 20$ нКл. На расстоянии $d = 5$ см от нити находится точечный заряд такой же величины, но противоположного знака.



1. Рассчитайте зависимость равнодействующей F сил, приложенных к бусинке, и потенциальной энергии системы E_p от координаты бусинки x в пределах от $x = -20$ см до $x = 20$ см с шагом 5 мм. За начало координат примите точку нити, ближайшую к точечному заряду $-q$. Результаты выведите в текстовый файл V4_F&E.DAT в виде таблицы:

```
  X,cm      F(X),mN      Ep(X),mJ

-20.0        0.0822     -0.01746
-19.5        0.0861     -0.01788
-19.0        0.0902     -0.01832
-18.5        0.0946     -0.01879
-18.0        0.0994     -0.01927
 ..............................
 18.0       -0.0994     -0.01927
 18.5       -0.0946     -0.01879
 19.0       -0.0902     -0.01832
 19.5       -0.0861     -0.01788
 20.0       -0.0822     -0.01746
```

2. Вследствие притяжения к точечному заряду бусинка совершает колебания относительно положения равновесия. Предположив, что бусинка отпущена без толчка из точки с координатой x_0, рассчитайте методом Эйлера-Кромера зависимости ее ускорения, скорости, координаты, а также полной механической энергии от времени для следующих трех случаев:

a. $x_0 = 2.5$ см;
b. $x_0 = 18$ см;
c. $x_0 = 18$ см, причем на бусинку дополнительно действует сила сопротивления, пропорциональная ее скорости, $F_c = -bv$, где $b = 1 \cdot 10^{-4}$ Н·с/м.

Расчеты выполните для интервала $0 \le t \le 4$ с, шаг вывода результатов 0.02 с, шаг метода Эйлера-Кромера $\Delta t = 0.00025$ с. Результаты выведите в текстовые файлы V4_A.DAT, V4_B.DAT, V4_C.DAT в виде таблиц вида:

```
t,s     a,m/s2     V,m/s      X,m      E,mJ

0.00    -0.994    0.0000    0.1800   -0.01927
0.02    -0.976   -0.0197    0.1798   -0.01927
0.04    -0.963   -0.0391    0.1792   -0.01927
0.06    -0.953   -0.0582    0.1782   -0.01928
0.08    -0.948   -0.0772    0.1769   -0.01929
0.10    -0.947   -0.0962    0.1751   -0.01930
.........................................
3.90    -3.389   -0.0490    0.0132   -0.06949
3.92    -2.967   -0.1131    0.0116   -0.06951
3.94    -2.242   -0.1658    0.0087   -0.06955
3.96    -1.228   -0.2010    0.0050   -0.06962
3.98    -0.030   -0.2139    0.0008   -0.06970
4.00     1.162   -0.2026   -0.0034   -0.06979
```

(здесь приведен фрагмент файла V4_C.DAT).

Пользуясь программой EASYPLOT, постройте графики:
– зависимостей $F(x)$ и $E_p(x)$ (два графика на экране);
– зависимостей $x(t)$ - график колебаний, $a(t)$ - график ускорения и $v(x)$ - фазовый портрет (по три графика на экране для каждого из случаев "a", "b", "c").

Подсказки

1. Как видно из рисунка в условии, равнодействующая **F** приложенных к бусинке сил складывается из кулоновской силы $\mathbf{F}_q$, силы реакции нити **N** и, возможно, силы сопротивления (не показана). Считая заряды точечными, по закону Кулона запишем:

$$F_q = k \cdot \frac{q^2}{r^2}, \text{ где } k = 9 \cdot 10^9 \text{ Н}\cdot\text{м}^2/\text{Кл}^2 .$$

Применяя теорему Пифагора в треугольнике OAB и учитывая, что этот треугольник подобен треугольнику, построенному на силах **F** и $\mathbf{F}_q$, имеем:

$$r = \sqrt{x^2 + d^2} \text{ , а также } F = F_q \cdot \frac{x}{r} .$$

Тогда модуль равнодействующей равен

$$F = kq^2 \cdot \frac{x}{\left(x^2 + d^2\right)^{\frac{3}{2}}} .$$

Перед этой формулой необходимо дописать знак минус, так как в уравнения движения входит не модуль, а проекция силы на координатную ось. Учитывая также силу сопротивления, окончательно получаем:

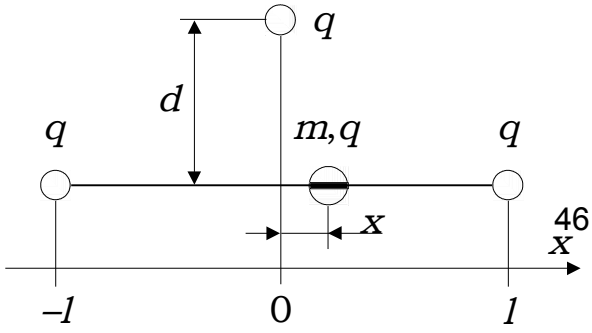
$$F(x,v) = -kq^2 \cdot \frac{x}{\left(x^2 + d^2\right)^{\frac{3}{2}}} - bv .$$

2. Потенциальная энергия взаимодействия бусинки с зарядом определяется выражением

$$E_p(x) = -\frac{kq^2}{\sqrt{x^2 + d^2}} .$$

*Вариант 5**

Параллельно горизонтальной оси OX натянута



гладкая диэлектрическая нить. В точках с координатами $-l$ и $l = 20$ см на ней находятся два точечных электрических заряда величиной $q = 20$ нКл. Третий заряд q расположен на расстоянии d от нити на равном удалении от первого и второго зарядов. По нити скользит небольшая бусинка массой $m = 100$ мг, также несущая заряд q. Помимо сил со стороны зарядов, на бусинку действует сила сопротивления, пропорциональная ее скорости, $F_c = -bv$, причем $b = 2 \cdot 10^{-5}$ Н·с/м.

1. Рассчитайте зависимость $F(x)$ результирующей силы, действующей на бусинку со стороны зарядов, (т.е. пока без учета силы сопротивления) от ее координаты для трех случаев:

a. $d = 50$ см;
b. $d = 12.5$ см;
c. $d = 8$ см.

Расчеты выполните в интервале $-15 \le x \le 15$ см с шагом 0.5 см. Результаты выведите в текстовый файл с именем V5_F.DAT в виде следующей таблицы:

X,cm	F_a,mN	F_b,mN	F_c,mN
-15.0	1.407	1.338	1.301
-14.5	1.156	1.085	1.045
-14.0	0.965	0.893	0.849
-13.5	0.816	0.742	0.694
.....			
13.5	-0.816	-0.742	-0.694
14.0	-0.965	-0.893	-0.849
14.5	-1.156	-1.085	-1.045
15.0	-1.407	-1.338	-1.301

Для тех же трех случаев рассчитайте зависимость потенциальной энергии системы $E_p(x)$ с выводом в файл V5_E.DAT таблицы:

X,cm	Ep(d1),mJ	Ep(d2),mJ	Ep(d3),mJ
-15.0	0.0892	0.1007	0.1035
-14.5	0.0828	0.0947	0.0976
-14.0	0.0775	0.0898	0.0929
-13.5	0.0731	0.0857	0.0891
.....			
13.5	0.0731	0.0857	0.0891
14.0	0.0775	0.0898	0.0929
14.5	0.0828	0.0947	0.0976

```
15.0     0.0892     0.1007     0.1035
```

2. Бусинку отвели на расстояние $x_0 = 15$ см от начала координат и в момент времени $t = 0$ отпустили без толчка. Методом Эйлера-Кромера рассчитайте зависимости ускорения, скорости, координаты и полной механической энергии бусинки для трех упомянутых случаев "a", "b", "c".

Расчеты выполните для интервала $0 \le t \le 10$ с, шаг вывода результатов 0.025 с, шаг метода Эйлера-Кромера $\Delta t = 0.0005$ с. Результаты выведите в текстовые файлы с именами V5_A.DAT, V5_B.DAT, V5_C.DAT в виде таблиц вида:

```
   t,s      a,m/s2      V,m/s       X,m       E,mJ

 0.000    -14.068     0.0000    0.1500    0.0892
 0.025    -11.783    -0.3316    0.1457    0.0891
 0.050     -7.798    -0.5769    0.1340    0.0889
 0.075     -4.750    -0.7318    0.1175    0.0887
 0.100     -2.855    -0.8254    0.0979    0.0885
 ..........................................
 9.900     -0.861     0.2481    0.0418    0.0479
 9.925     -0.998     0.2249    0.0478    0.0479
 9.950     -1.129     0.1983    0.0530    0.0478
 9.975     -1.251     0.1686    0.0576    0.0478
10.000     -1.358     0.1359    0.0614    0.0478
```

(здесь приведен фрагмент файла V5_A.DAT).

Пользуясь программой EASYPLOT, постройте графики:
- зависимостей $F(x)$ и $E_p(x)$ (два графика на экране по три кривых на каждом для случаев "a", "b" и "c");
- зависимостей $x(t)$ - график колебаний, $a(t)$ - график ускорения и $v(x)$ - фазовый портрет (по три графика на экране для каждого из случаев "a", "b" и "c").

Подсказки

1. Данная задача является комбинацией задач вариантов 3 и 4, с той разницей, что заряд, расположенный вне нити, не притягивает, а *отталкивает* бусинку. Это приводит к интересным особенностям в поведении системы.
2. Проанализируйте рисунки и подсказки к задачам вариантов 3 и 4. После этого для результирующей силы нетрудно будет записать:

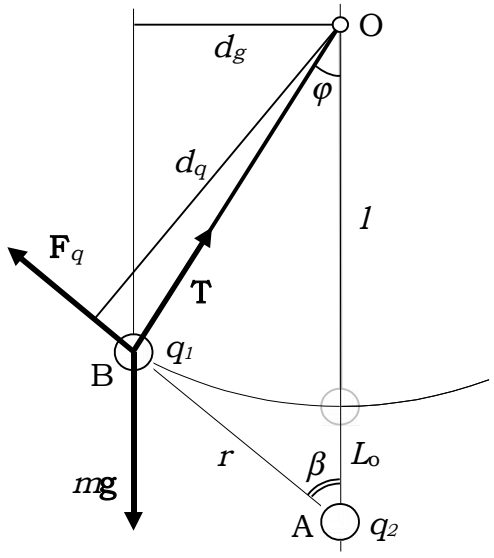
$$F(x,v) = kq^2 \cdot \left[\frac{1}{\left(x^2 + d^2\right)^{\frac{3}{2}}} - \frac{4l}{\left(l^2 - x^2\right)^2} \right] \cdot x - bv \ .$$

3. Выражение для потенциальной энергии бусинки

$$E_p(x) = kq^2 \cdot \left(\frac{1}{\sqrt{x^2 + d^2}} + \frac{2l}{l^2 - x^2} \right) .$$

4. В Pascal-программе при описании функций результирующей силы и потенциальной энергии к списку их аргументов следует добавить параметр *d*: $F(x,v,d)$, $E_p(x,d)$. Это даст возможность в одном цикле легко сформировать таблицы, рассчитываемые в пункте 1 задания.

*Вариант 6**



Маятник представляет собой небольшой шарик массой $m = 0.5$ г, прикрепленный к невесомому диэлектрическому стержню длиной $l = 20$ см, второй конец которого шарнирно закреплен. Шарик несет электрический заряд $q_1 = +100$ нКл. Под точкой подвеса маятника на расстоянии $L_0 = 10$ см от нижнего положения шарика расположен точечный заряд q_2. На маятник действует сила сопротивления, причем момент этой силы пропорционален угловой скорости движения маятника: $M_c = -b\omega$, где $b = 10^{-5}$ Н·м·с/рад. Ускорение свободного падения $g = 9.81$ м/с^2.

1. Рассчитайте зависимость результирующего момента *M* силы тяжести и кулоновской силы от угла отклонения маятника φ для трех случаев:

a. $q_2 = -100$ нКл;

b. $q_2 = 0$;

d. $q_2 = +100$ нКл.

Расчеты выполните в интервале изменения угла от $\varphi = -3$ рад до $\varphi = 3$ рад с шагом 0.05 рад. Результаты выведите в

текстовый файл с именем V6_M.DAT в виде следующей таблицы:

Phi,rad	M_a,mN*m	M_b,mN*m	M_c,mN*m
-3.00	0.145	0.138	0.132
-2.95	0.195	0.187	0.178
-2.90	0.245	0.235	0.224
-2.85	0.295	0.282	0.269
.....			
2.85	-0.295	-0.282	-0.269
2.90	-0.245	-0.235	-0.224
2.95	-0.195	-0.187	-0.178
3.00	-0.145	-0.138	-0.132

Для тех же трех случаев рассчитайте зависимость потенциальной энергии системы $E_p(\varphi)$ с выводом в файл V6_E.DAT таблицы:

Phi,rad	Ep_a,mJ	Ep_b,mJ	Ep_c,mJ
-3.00	1.772	1.952	2.133
-2.95	1.763	1.944	2.125
-2.90	1.752	1.934	2.115
-2.85	1.739	1.921	2.102
.....			
2.85	1.739	1.921	2.102
2.90	1.752	1.934	2.115
2.95	1.763	1.944	2.125
3.00	1.772	1.952	2.133

2. Маятник отклонили на угол $\varphi_0 = 3.0$ рад и в начальный момент времени отпустили без толчка. Методом Эйлера-Кромера рассчитайте зависимости угловой скорости ω, угла поворота φ маятника, а также его полной механической энергии для трех упомянутых случаев "a", "b", "c".

Расчеты выполните для интервала $0 \le t \le 5$ с, шаг вывода результатов 0.02 с, шаг метода Эйлера-Кромера $\Delta t = 0.0005$ с. Результаты выведите в текстовые файлы с именами V6_A.DAT, V6_B.DAT, V6_C.DAT в виде таблиц вида:

t,s	Omega,rad/s	Phi,rad	E,mJ
0.00	0.000	3.000	1.772
0.02	-0.144	2.999	1.772
0.04	-0.290	2.994	1.772
0.06	-0.440	2.987	1.772
0.08	-0.598	2.976	1.772

```
0.10      -0.766       2.963       1.772
.....................................
4.90       6.732      -0.123      -0.401
4.92       7.001       0.015      -0.409
4.94       6.449       0.151      -0.417
4.96       5.405       0.270      -0.424
4.98       4.221       0.366      -0.429
5.00       3.029       0.438      -0.432
```

(здесь приведен фрагмент файла V6_A.DAT).

Пользуясь программой EASYPLOT, постройте графики:
- зависимостей $M(\varphi)$ и $E_p(\varphi)$ (два графика на экране по три кривых на каждом для случаев "a", "b" и "c");
- зависимостей $\varphi(t)$ - график колебаний, $\omega(t)$ - график угловой скорости и $\omega(\varphi)$ - фазовый портрет (по три графика на экране для каждого из случаев "a", "b" и "c").

Подсказки

1. На шарик маятника действуют сила тяжести $m\mathbf{g}$, кулоновская сила $\mathbf{F}_q$, сила реакции стержня $\mathbf{T}$ и сила сопротивления (на рисунке в условии не показана). Результирующий момент этих сил относительно оси вращения O равен

   $$M = -mg \cdot d_g + F_q \cdot d_q + 0 - b\omega \quad ,$$

   где d_g, d_q – плечи соответствующих сил. Все входящие сюда величины необходимо выразить через угол отклонения маятника φ.
2. Анализируя рисунок, получаем

   $$d_g = l \cdot \sin\varphi \, , \quad d_q = (l + L_0) \cdot \sin\beta \quad .$$

   Согласно теореме синусов для треугольника OAB

   $$\sin\beta = \frac{l}{r(\varphi)} \cdot \sin\varphi \quad .$$

   По закону Кулона

   $$F_q = k \cdot \frac{q_1 \cdot q_2}{r^2(\varphi)} \quad , \quad \text{где} \quad k = 9 \cdot 10^9 \ \text{Н}\cdot\text{м}^2/\text{Кл}^2.$$

   Обозначение $r(\varphi)$ подчеркивает, что r зависит только от угла φ.

   Подставляя эти величины в выражение для результирующего момента силы, получаем окончательно:

$$M(\varphi,\omega) = -mgl\cdot\sin\varphi + \left[\frac{k q_1 q_2 (l+L_0)}{r^3(\varphi)}\right]\cdot l\sin\varphi - b\omega \quad .$$

3. Потенциальная энергия системы складывается из потенциальной энергии шарика в поле силы тяжести и энергии взаимодействия зарядов:

$$E_p(\varphi) = mgl\cdot(1-\cos\varphi) + k\cdot\frac{q_1\cdot q_2}{r(\varphi)} \quad .$$

4. Остается найти зависимость $r(\varphi)$. Применим теорему косинусов к стороне АВ треугольника ОАВ:

$$r(\varphi) = \sqrt{l^2 + (l+L_0)^2 - 2l\cdot(l+L_0)\cdot\cos\varphi} \quad .$$

Имеет смысл ввести безразмерный параметр $\xi = l/L_0$. Тогда после несложных преобразований можно получить:

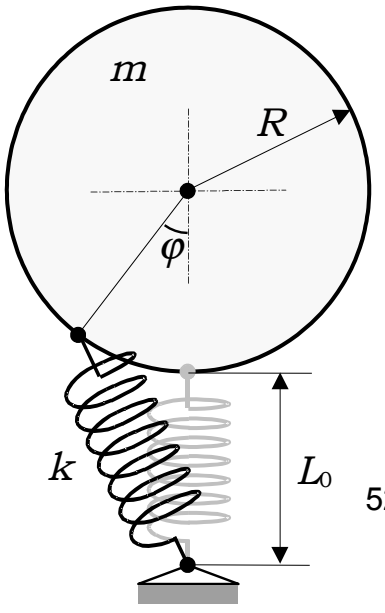
$$r(\varphi) = L_0\sqrt{1 + 2\xi(1+\xi)(1-\cos\varphi)} \quad .$$

Эта зависимость входит в несколько формул, поэтому ее целесообразно описать как самостоятельную функцию в Pascal-программе. Кроме того, при описании в программе функций результирующего момента силы и потенциальной энергии к списку их аргументов следует добавить заряд q_2: $M(\varphi,\omega,q_2)$, $E_p(\varphi,q_2)$. Это дает возможность в одном цикле легко сформировать таблицы, рассчитываемые в пункте 1 задания.

Поскольку ξ является константой, формулу для ее вычисления можно поместить прямо в разделе описания констант программы.

*Вариант 7**

Маховик в виде однородного сплошного диска массой $m = 1$ кг и радиусом $R = 0.1$ м способен вращаться без трения вокруг проходящей через центр оси. Край маховика шарнирно соединен с пружиной жесткостью $k = 20$ Н/м, второй конец которой шарнирно



закреплен на расстоянии $L_0 = 0.15$ м от края маховика. Длина нерастянутой пружины $l_s = 0.15$ м.

1. Рассчитайте зависимость момента силы M, приложенной к маховику со стороны пружины, и потенциальной энергии деформации пружины E_p от угла поворота маховика φ в пределах от $\varphi = -5$ рад до $\varphi = 5$ рад с шагом 0.1 рад. За начальное $(\varphi = 0)$ примите положение маховика, при котором длина пружины минимальна (см. рис.). Результаты выведите в текстовый файл с именем V7_M&E.DAT в виде таблицы следующего вида:

Phi,rad	M(Phi),mN*m	Ep(Phi),J
-5.0	-181.6	0.0837
-4.9	-198.1	0.1027
-4.8	-211.8	0.1232
-4.7	-222.6	0.1450
-4.6	-230.2	0.1676
-4.5	-234.3	0.1909
.....		
4.5	234.3	0.1909
4.6	230.2	0.1676
4.7	222.6	0.1450
4.8	211.8	0.1232
4.9	198.1	0.1027
5.0	181.6	0.0837

2. Пусть в начальный момент времени $\varphi = 0$, и маховику сообщена начальная угловая скорость ω_0. Методом Эйлера-Кромера рассчитайте зависимости углового ускорения ε, угловой скорости ω, угла поворота φ маховика, а также полной механической энергии системы маховик-пружина от времени для следующих трех случаев:

a. $\omega_0 = 2.0$ рад/с;

b. $\omega_0 = 12.5$ рад/с;

c. $\omega_0 = 13.0$ рад/с.

Расчеты выполните для интервала $0 \le t \le 5$ с, шаг вывода результатов 0.02 с, шаг метода Эйлера-Кромера $\Delta t = 0.0005$ с. Результаты выведите в текстовые файлы с именами V7_A.DAT, V7_B.DAT, V7_C.DAT в виде таблиц следующего вида:

t,s	Eps,rad/s2	Omega,rad/s	Phi,rad	E,mJ
0.00	0.00	2.000	0.000	10.00

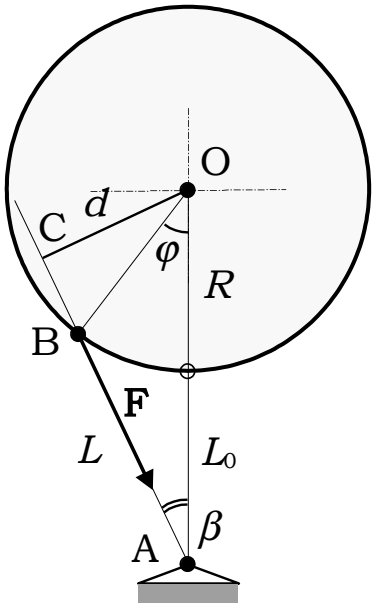
```
0.02       -0.00       2.000       0.040       10.00
0.04       -0.03       2.000       0.080       10.00
0.06       -0.09       1.999       0.120       10.00
0.08       -0.22       1.996       0.160       10.00
0.10       -0.42       1.989       0.200       10.00
.....................................................
4.90       10.55       0.257      -0.656       10.00
4.92       10.28       0.465      -0.649        9.99
4.94        9.87       0.667      -0.638        9.99
4.96        9.32       0.859      -0.622        9.99
4.98        8.65       1.040      -0.603        9.99
5.00        7.90       1.205      -0.581        9.99
```

(здесь приведен фрагмент файла V7_A.DAT).

Пользуясь программой EASYPLOT, постройте графики:
- зависимостей $M(\varphi)$ и $E_p(\varphi)$ (два графика на экране);
- зависимостей $\varphi(t)$ - график колебаний, $\varepsilon(t)$ - график углового ускорения и $\omega(\varphi)$ - фазовый портрет (по три графика на экране для каждого из случаев a, b, c).

Подсказки

5. Прежде всего, необходимо выразить момент силы, действующей на маховик, и потенциальную энергию деформированной пружины от угла поворота маховика. Для этого перестроим приведенный в условии рисунок, указав силу упругости **F** и ее плечо *d*.



6. Обозначим длину растянутой пружины через *L*, тогда по закону Гука $F=-k\cdot(L-l_s)$, а момент силы $M=-k\cdot(L-l_s)\cdot d$. Плечо найдем из треугольника OAC: $d=(R+L_0)\cdot\sin\beta$. Применяя теорему синусов к треугольнику OAB, получим $\sin\beta=R\cdot\sin\varphi/L$. Тогда

$$M(\varphi)=-k\cdot\left(1-\frac{l_s}{L(\varphi)}\right)\cdot(R+L_0)\cdot R\cdot\sin\varphi$$

Здесь обозначение $L(\varphi)$ подчеркивает, что *L* зависит только от угла φ.

7. Из формулы для потенциальной энергии упруго деформированного тела имеем

$$E_p(\varphi) = \frac{k}{2} \cdot [\, L(\varphi) - l_s]^2 \ .$$

8. Для нахождения зависимости $L(\varphi)$ применим теорему косинусов к стороне АВ треугольника ОАВ:

$$L(\varphi) = \sqrt{R^2 + (R + L_0)^2 - 2R(R + L_0) \cdot \cos\varphi} \ .$$

Здесь оказывается удобным ввести безразмерный параметр $\xi = R/L_0$. С учетом этого обозначения после несложных преобразований можно получить:

$$L(\varphi) = L_0 \sqrt{1 + 2\xi(1+\xi)(1-\cos\varphi)}$$

Эта зависимость входит в несколько формул, поэтому ее целесообразно описать как самостоятельную функцию в Pascal-программе.

9. Используя величину ξ, можно выражение для момента силы преобразовать к компактной окончательной форме:

$$M(\varphi) = -A \cdot \left(1 - \frac{l_s}{L(\varphi)}\right) \ , \text{ где } \ A = k L_0^2 \xi(1+\xi) \ .$$

Поскольку ξ и A являются константами, формулы для их вычисления можно поместить прямо в разделе описания констант программы.

10. Формула для момента инерции сплошного однородного диска относительно оси, проходящей через центр перпендикулярно его плоскости, имеет вид: $I = m R^2/2$.
11. Полная энергия системы складывается из потенциальной энергии деформации пружины и кинетической энергии вращения маховика:

$$E = E_p(\varphi) + \frac{I\omega^2}{2} \ .$$

*Вариант 8**

Маховик в виде однородного сплошного диска массой $m = 1$ кг и радиусом $R = 0.1$ м способен вращаться вокруг

проходящей через центр оси (см. рисунок к задаче 7-го варианта). Край маховика шарнирно соединен с пружиной жесткостью $k = 20$ Н/м, второй конец которой шарнирно закреплен на расстоянии $L_0 = 0.15$ м от края маховика. Длина нерастянутой пружины $l_s = 0.21$ м.

1. Рассчитайте зависимость момента силы M, приложенной к маховику со стороны пружины, и потенциальной энергии деформации пружины E_p от угла поворота маховика φ в пределах от $\varphi = -4$ рад до $\varphi = 4$ рад с шагом 0.1 рад. За начальное $(\varphi = 0)$ примите положение маховика, при котором длина пружины минимальна. Результаты выведите в текстовый файл с именем V8_M&E.DAT в виде таблицы следующего вида:

Phi,rad	M(Phi),mN*m	Ep(Phi),J
-4.0	-133.4	0.1307
-3.9	-124.9	0.1436
-3.8	-114.0	0.1556
-3.7	-100.8	0.1663
-3.6	-85.6	0.1757
-3.5	-68.8	0.1834
.....	.....	.....
3.5	68.8	0.1834
3.6	85.6	0.1757
3.7	100.8	0.1663
3.8	114.0	0.1556
3.9	124.9	0.1436
4.0	133.4	0.1307

2. Пусть в начальный момент времени $\varphi = 0.1$ рад, и маховику сообщена начальная угловая скорость ω_0. Методом Эйлера-Кромера рассчитайте зависимости углового ускорения ε, угловой скорости ω, угла поворота φ маховика, а также полной механической энергии системы маховик-пружина от времени для следующих трех случаев:

a. $\omega_0 = 0.5$ рад/с;
b. $\omega_0 = 1.5$ рад/с;
c. $\omega_0 = 10.0$ рад/с, причем на маховик в этом случае дополнительно действует момент силы вязкого трения, пропорциональный его угловой скорости: $M_{тр} = -b\omega$, где $b = 0.0016$ Н·м·с/рад.

Расчеты выполните для интервала $0 \le t \le 10$ с, шаг вывода результатов 0.05 с, шаг метода Эйлера-Кромера $\Delta t = 0.001$ с.

Результаты выведите в текстовые файлы с именами V8_A.DAT, V8_B.DAT, V8_C.DAT в виде таблиц следующего вида:

t,s	Eps,rad/s2	Omega,rad/s	Phi,rad	E,mJ
0.00	0.72	10.000	0.100	285.01
0.05	7.23	10.313	0.607	276.63
0.10	-9.61	10.314	1.126	268.44
0.15	-26.30	9.383	1.621	260.94
0.20	-30.98	7.903	2.054	254.92
.....				
9.80	10.67	0.527	6.868	12.58
9.85	9.90	1.044	6.908	12.51
9.90	8.45	1.507	6.972	12.37
9.95	6.07	1.876	7.057	12.14
10.00	2.74	2.101	7.158	11.84

(здесь приведен фрагмент файла V8_C.DAT).

Пользуясь программой EASYPLOT, постройте графики:
- зависимостей $M(\varphi)$ и $E_p(\varphi)$ (два графика на экране);
- зависимостей $\varphi(t)$ - график колебаний, $\varepsilon(t)$ - график углового ускорения и $\omega(\varphi)$ - фазовый портрет (по три графика на экране для каждого из случаев a, b, c).

Подсказки

1. Внимательно изучите подсказки к задаче 7-го варианта. Отличие Вашего задания в том, что $L_0 < l_s$, т.е. в определенном интервале углов поворота маховика пружина оказывается *сжатой*. Точка $\varphi = 0$ оказывается положением *неустойчивого* равновесия, и это приводит к интересным особенностям в поведении системы.
2. Наличие вязкого трения приводит к появлению дополнительного слагаемого в выражении для результирующего момента силы. Теперь

$$M(\varphi,\omega) = -A \cdot \left(1 - \frac{l_s}{L(\varphi)}\right) - b\omega \quad .$$

При написании программы имеет смысл сразу запрограммировать самый сложный случай "c", а для случаев "a" и "b" запускать программу, полагая $b = 0$.

* * *

Глава 3

Решение нелинейных уравнений численными методами

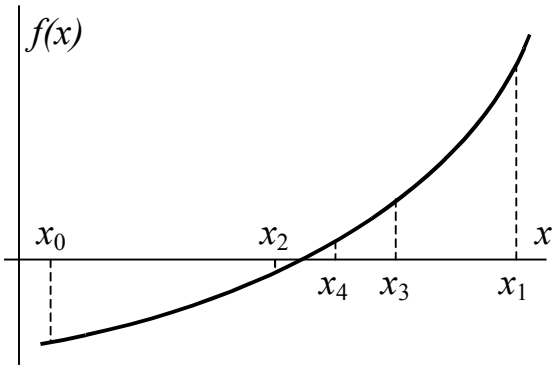
Если график зависимости одной величины от другой не является прямой линией, то такая зависимость называется нелинейной. В физике такие зависимости встречаются очень часто, и тогда при решении задач приходится сталкиваться с уравнениями, которые называют нелинейными. Их всегда можно представить в стандартном виде

$$f(x) = 0 \qquad (1)\ .$$

Простейшее нелинейное уравнение – обычное квадратное уравнение. Однако для более сложных уравнений получить решение в аналитическом виде (т.е. в виде формулы) оказывается невозможным. Тогда используют графические или численные методы.

Существует свыше десятка разных методов численного решения уравнений вида (1). Мы рассмотрим два наиболее простых и удобных для решения физических задач метода: дихотомия и метод секущих.

1. Дихотомия (метод половинного деления)



Это наиболее надежный метод решения уравнений. Он может обеспечить максимальную точность нахождения корня, равную точности, с которой представимы числа в компьютере.

Допустим, уравнение записано в стандартном виде (1), а график функции $f(x)$ имеет вид, показанный на

рисунке. Предположим, известно, что внутри отрезка $[x_0, x_1]$ находится ровно один корень уравнения. Тогда функция $f(x)$ на концах отрезка имеет разные знаки[1]. Иными словами, $f(x_0) \cdot f(x_1) < 0$. Найдем середину отрезка $x_2 = (x_0 + x_1)/2$ и вычислим $f(x_2)$. Далее из двух половин исходного отрезка $[x_0, x_2]$ и $[x_2, x_1]$ выберем ту, для которой выполняется условие $f(x_2) \cdot f(x_{\text{гран.}}) < 0$, ибо корень лежит именно на этой половине (на рисунке – внутри отрезка $[x_2, x_1]$). Затем этот отрезок снова делим пополам и получаем точку x_3 и т.д. Таким образом, с каждым шагом описанной процедуры длина отрезка, внутри которого заключен корень, становится вдвое меньше. Если требуется найти корень с точностью ε, то деление пополам нужно продолжать до тех пор, пока длина отрезка не станет меньше, либо равна ε. Тогда любая точка последнего отрезка даст значение корня с требуемой точностью.

Скорость процесса поиска корня сравнительно невелика: для уточнения одной значащей десятичной цифры требуется примерно три цикла половинного деления. Зато точность ответа гарантируется.

Метод дихотомии имеет следующие недостатки. Невозможно предсказать, к какому из корней сойдется процесс поиска, если исходный отрезок $[x_0, x_1]$ содержит несколько корней. Метод не применим для поиска корней четной кратности, поскольку в этом случае кривая $f(x)$ не пересекает, а касается оси абсцисс, и функция справа и слева вблизи корня имеет одинаковые знаки.

2. Программная реализация метода дихотомии

Простейшая программа, реализующая метод дихотомии, – приведенная ниже программа EQU1. Она проводит решение уравнения (1) с функцией $f(x)$ вида

$$f(x) = 2x \cdot \sin x - \cos x \qquad (2).$$

Эта функция описана в самом начале программы под именем `F`.

Раздел операторов программы предельно прост и состоит всего из двух процедур: `Writeln`, выводящей на экран значение корня, и `Readln`, приостанавливающей

[1] Предполагается, что корень не является кратным.

работу программы до нажатия клавиши Enter. Поиск нуля функции (2) осуществляется в теле функции Root, которая в нашем случае вычисляется непосредственно при выполнении процедуры Writeln.

Остановимся подробнее на работе функции Root. Она имеет три параметра: X0, X1 - границы интервала, в которых ищется корень, и E - точность нахождения корня. При вызове функции им присваиваются значения 0.5, 1 и 0.0001, соответственно. В функции используется локальная переменная X, в которой хранится текущее значение корня.

Основу функции образует цикл repeat..until, начинающийся с присвоения X значения из середины интервала поиска корня. Затем сравниваются знаки функции F на левой границе интервала X0 и в его середине X. В случае, если знаки различны, это означает что корень расположен левее точки X, поэтому точка X принимается за новую правую границу интервала поиска корня. В противном случае, т.е. когда знаки функции одинаковы, точка X принимается за новую левую границу, поскольку корень расположен правее этой точки. Далее следует сравнение длины интервала поиска корня с заданной точностью вычисления и, если она достигнута, происходит выход из цикла с присвоением значения X функции Root. В противном случае следует возврат на начало цикла и присвоение X нового текущего значения.

Выход из цикла происходит также при равенстве функции F нулю, которое возможно при случайном точном попадании в корень при очередном делении интервала пополам.

```
program EQU1;

function F (x:real): real;
begin
  F:= 2*x*Sin(x) - Cos(x)
end;  { F }

function Root (X0, X1, E: real): real;
      { возвращает корень уравнения F(x)=0, }
      {     найденный методом дихотомии     }
var
  X: real;
begin
  repeat
    X:= (X0+X1)/2;
```

```
    if F(X0)*F(X) < 0 then
      X1:= X
    else
      X0:= X
  until (Abs (X1-X0) < E) or (F (X) = 0);
  Root:= X;
end;  { Root }

BEGIN
  Writeln ('Корень равен', Root (0.5, 1, 0.0001):8:4);
  Readln
END.

Корень равен  0.6533
```

Программа EQU1 имеет существенный недостаток. Он состоит в том, что при каждом выполнении цикла половинного деления производится *три* вычисления функции `F`, хотя в рассмотрение всякий раз включается только *одна* новая точка `X`.

Недостаток устранен в программе EQU2. Здесь для исключения ненужных повторных вычислений функций в одной и той же точке в `Root` используются еще две вспомогательные локальные переменные `U` и `Y`. Первой присваивается значение функции `F(X0)` перед входом в цикл, а второй – `F(X)` сразу после деления отрезка пополам внутри цикла. В дальнейшем эти значения используются там, где в программе EQU1 происходило обращение к функции `F`. При решении сложных уравнений такой прием ускоряет работу программы почти в 3 раза.

```
program EQU2;

function F (x:real): real;
begin
  F:= 2*x*Sin(x) - Cos(x)
end;  { F }

function Root (X0, X1, E: real): real;
      { возвращает корень уравнения F(x)=0, }
      {     найденный методом дихотомии     }
var
  X, Y, U: real;
begin
  U:= F(X0);
  repeat
    X:= (X0+X1)/2;
```

```
    Y:= F(X);
    if U*Y < 0 then
      X1:= X
    else
      begin
        X0:= X;
        U:= Y
      end
  until (Abs (X1-X0) < E) or (Y = 0);
  Root:= X;
end;  { Root }

BEGIN
  Writeln ('Корень равен', Root (0.5, 1, 0.0001):8:4);
  Readln
END.

Корень равен  0.6533
```

Программу решения уравнений методом дихотомии нетрудно видоизменить так, чтобы она находила корни с наивысшей точностью, с которой могут быть представлены числа в данном компьютере. Для этого достаточно изменить признак, по которому происходит выход из цикла половинного деления.

Числа в формате `real` представляются в компьютере с точностью 11-12 значащих цифр. При многократном делении отрезка $[x_0, x_1]$ пополам значения x_0 и x_1 в конце концов будут различаться лишь в последней значащей цифре. Поэтому при вычислении середины отрезка $x=(x_0+x_1)/2$ рано или поздно компьютер выдаст результат $x=x_0$ или $x=x_1$. Это означает, что достигнута предельная точность вычисления корня, и выход из цикла можно организовать по признакам $x=x_0$ и $x=x_1$.

Достигнуть еще более высокой точности можно, используя вместо типа `real` тип `extended`. Он отличается повышенной точностью представления чисел в компьютере (19 – 20 значащих цифр вместо 11 – 12) и расширенным диапазоном значений (10^{-4951} – 10^{4932} вместо 10^{-39} – 10^{38}). Имеющийся во многих компьютерах математический сопроцессор[1], позволяющий иногда ускорить вычисления в десятки раз, создан как раз для работы с числами типа

[1] Специальная микросхема, выполняющая математические операции схемным (а не программным) путем. В современных компьютерах сопроцессор встроен в микросхему центрального процессора

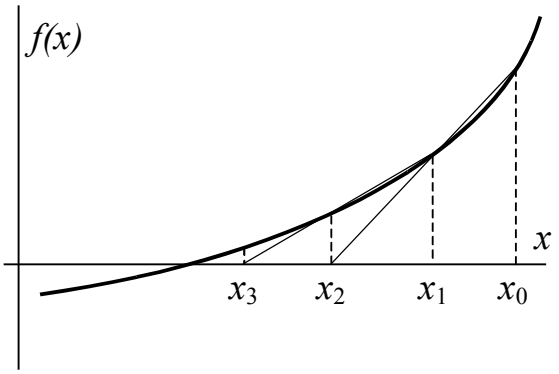
extended. Для подключения сопроцессора к вычислениям используется специальная директива Turbo-Pascal, которая имеет вид {$N+} и располагается в начале программы (см. приведенный ниже пример). Там же обычно стоит директива E+, означающая, что если компьютер не имеет микросхемы сопроцессора, то ее работа будет автоматически эмулирована (имитирована) программным путем (конечно, без увеличения скорости вычислений).

Ниже приводится программа EQU3 для решения уравнения с предельно высокой точностью и результат ее работы.

```
program EQU3;
      {$N+,E+}

function F (x:extended): extended;
begin
  F:= 2*x*Sin(x) - Cos(x)
end;  { F }

function Root (X0, X1: extended): extended;
      { возвращает корень уравнения F(x)=0, }
      {     найденный методом дихотомии     }
      {       с максимальной точностью      }
var
  X, Y, U: extended;
begin
  U:= F(X0);
  repeat
    X:= (X0+X1)/2;
    Y:= F(X);
    if (X0=X) or (X1=X) then
      Y:= 0;
    if U*Y < 0 then
      X1:= X
    else
      begin
        X0:= X;
        U:= Y
      end
  until Y = 0;
  Root:= X;
end;  { Root }

BEGIN
  Writeln ('Корень равен', Root (0.5, 1):21:20);
  Readln
END.
```

```
Корень равен 0.6532711870944031O1
```

3. Метод секущих



В отличие от метода дихотомии, метод секущих позволяет находить кратные корни уравнения (1). К тому же скорость сходимости этого метода значительно выше, т.е. та же точность может быть получена при меньшем числе итераций (циклов последовательных приближений).

Пусть дано уравнение (1), а график функции $f(x)$ имеет вид, изображенный на рисунке.

Выберем две точки x_0 и x_1 вблизи корня и проведем секущую через точки $f(x_0)$ и $f(x_1)$ до пересечения с осью абсцисс. Точку пересечения обозначим x_2 и будем считать новым приближением корня. Далее проводим секущую через точки $f(x_1)$ и $f(x_2)$ и получаем более точное значение x_3 и т.д. Этот процесс продолжаем до тех пор, пока не будет выполнено соотношение

$$|x_{n+1} - x_n| < \varepsilon \qquad (3),$$

где ε - точность вычислений. Тогда x_{n+1} будет искомым значением корня уравнения.

Найдем значение x_2 по заданным x_0 и x_1. Для этого рассмотрим подобные треугольники, образованные секущими, осью абсцисс и высотами, опущенными из точек пересечения секущих с графиком функции. Как видно из рисунка,

$$\frac{f(x_0)}{x_0 - x_2} = \frac{f(x_1)}{x_1 - x_2}, \quad \text{откуда} \quad x_2 = x_1 - \frac{(x_1 - x_0)\cdot f(x_1)}{f(x_1) - f(x_0)} .$$

В общем виде эту формулу можно записать

$$x_{n+1} = x_n - \frac{(x_n - x_{n-1})\cdot f(x_n)}{f(x_n) - f(x_{n-1})} \qquad (4).$$

Таким образом, нахождение корня сводится к двум операциям:

- Задание двух начальных приближений в окрестности корня,
- Последовательное применение формулы (4) до тех пор, пока не будет выполнено условие (3).

4. Программная реализация метода секущих

Решение уравнения (1) с функцией (2) методом секущих используется в программе EQU4. Как и в предыдущих рассмотренных программах, алгоритм поиска корня реализован в виде функции `Root`. Ее параметры: `X0` и `X1` - начальные приближения корня, `E` - точность. В нашем случае параметры функции `Root` равны, соответственно, 0.5, 1, 0.0001 .

Для того, чтобы избежать многократных вычислений функции `F` в одной и той же точке, используются вспомогательные локальные переменные `U` и `Y`. Кроме того, второе слагаемое в формуле (4) обозначено идентификатором `Delta`.

Основу функции `Root` образует цикл последовательных приближений к корню `repeat..until`. Перед входом в него вычисляется значение `F(X0)`, а сразу после входа - значение `F(X1)`, т.е. подготавливаются значения для вычисления следующего приближения корня. Однако вместо прямого использования формулы (4) целесообразно сначала вычислить ее второе слагаемое `Delta`. Дело в том, что после вычисления `Delta` "старые" значения `X0` и `U=F(X0)` сохранять больше не нужно, и им можно присвоить "новые" значения `X1` и `Y=F(X1)`, которые в следующем цикле будут исполнять роль "старых". Лишь после такого переприсваивания вычисляется следующее приближение корня и записывается на место предыдущего приближения. Если после этого заданная точность оказывается достигнутой, функции `Root` присваивается полученное значение корня и, вычисления завершаются. В противном случае цикл выполняется снова.

```
program EQU4;

function F (x:real): real;
begin
  F:= 2*x*Sin(x) - Cos(x)
end;  { F }

function Root (X0, X1, E: real): real;
      { возвращает корень уравнения F(x)=0, }
      {      найденный методом секущих      }
var
  Y, U, Delta: real;
begin
  U:= F(X0);
  repeat
    Y:= F (X1);
    Delta:= (X1-X0)*Y/(Y-U);
    X0:= X1;
    U:= Y;
    X1:= X1-Delta
  until Abs (Delta) < E;
  Root:= X1;
end;  { Root }

BEGIN
  Writeln ('Корень равен', Root (0.5, 1, 0.0001):8:4);
  Readln
END.

Корень равен  0.6533
```

Расчет по программе EQU4 занимает приблизительно в 3 раза меньшее время, чем вычисление корня с той же точностью методом дихотомии по программе EQU2. Поэтому более быстрый метод секущих предпочтительнее использовать в тех случаях, когда требуется многократное вычисление корня.

Часто уравнение содержит параметр, который при решении задачи меняется в некоторых пределах с заданным шагом. Тогда организуется цикл, в котором уравнение решается для каждого значения параметра.

Пусть, например, необходимо решить уравнение

$$f(x) = p \cdot x \cdot \sin x - \cos x \qquad (5)$$

для 6 значений параметра p, изменяющегося от 0 до 2.5 с шагом 0.5. Иными словами, требуется рассчитать значения функции $x = x(p)$, заданной неявно.

Используя в качестве основы программу EQU4, нетрудно составить программу EQU5 для решения этой задачи.

```
program EQU5;

var
  X, p: real;

function F (x:real): real;
begin
  F:= p*x*Sin(x) - Cos(x)
end;  { F }

function Root (X0, X1, E: real): real;
      { возвращает корень уравнения F(x)=0, }
      {      найденный методом секущих      }
var
  Y, U, Delta: real;
begin
  U:= F(X0);
  repeat
    Y:= F (X1);
    Delta:= (X1-X0)*Y/(Y-U);
    X0:= X1;
    U:= Y;
    X1:= X1-Delta
  until Abs (Delta) < E;
  Root:= X1;
end;  { Root }

BEGIN
  p:= 0; X:= 1.5;
  while p < 2.6 do
    begin
      X:= Root (X, 1.01*X, 0.0001);
      Writeln (p:5:1, X:8:4);
      p:= p+0.5
    end;
  Readln
END.
```

```
  0.0  1.5708
  0.5  1.0769
  1.0  0.8603
  1.5  0.7360
  2.0  0.6533
  2.5  0.5932
```

Особенностью данной программы является то, что перед началом процесса поиска корня задается только одно его начальное приближение `X`. Для начального значения

параметра $p = 0$ его легко определить непосредственно из уравнения (5): при $p = 0$ $x = \pi/2 \approx 1.5$. В качестве второго начального приближения используется первое, помноженное на 1.01.

Еще один красивый прием: в каждом следующем цикле с новым значением параметра p в качестве нового начального приближения x используется значение корня, полученное для предыдущего значения параметра. Благодаря этому, во-первых, не приходится гадать, где находится корень для $p \neq 0$, и, во-вторых, при изменении p с небольшим шагом начальное приближение в следующем цикле всегда оказывается вблизи корня.

ТРЕНИРОВОЧНЫЕ ПРИМЕРЫ

Напишите и отладьте на компьютере три программы для решения предложенного в Вашем варианте уравнения:

– методом дихотомии с точностью 0.0001;
– методом дихотомии с максимально достижимой точностью;
– методом секущих с точностью 0.0001.

Значение корня выведите на экран компьютера.

Подсказка:

Данные тренировочные примеры рассматривайте как "полигон" для отладки функции `Root`. В дальнейшем она без изменений сможет быть перенесена в любую программу в виде готового блока. Такой подход позволит значительно облегчить написание сложных программ, выполняющих решение реальных задач.

Вар.	Уравнение	$[x_0 ; x_1]$	Ответ
1	$x + \sqrt{x} + \sqrt[3]{x} = 2.5$	[0.4 ; 1]	0.737619246277462784
2	$\mathrm{tg}\, x - \frac{1}{3}\mathrm{tg}^3 x + \frac{1}{5}\mathrm{tg}^5 x = \frac{1}{3}$	[0 ; 0.8]	0.333255464672049843

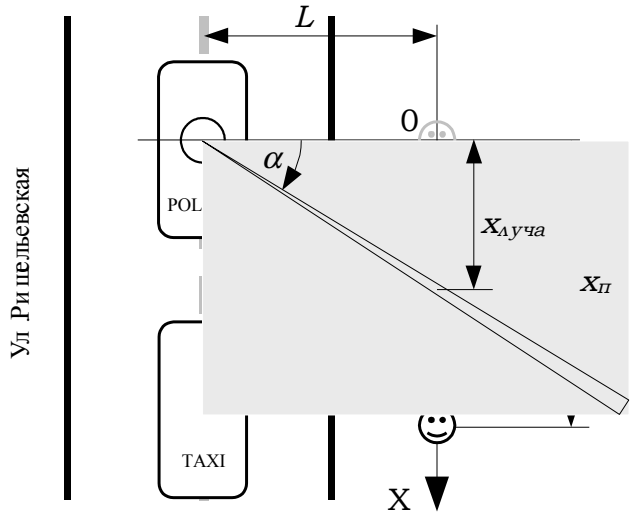
3	$3x+e^{x}-e^{-x}=14$	[1 ; 3]	2.069218196373615370
4	$\sqrt{1-x}=\operatorname{tg}x$	[0 ; 1]	0.576769807570751599
5	$x+\cos\left(x^{0.52}+2\right)=0$	[0.5 ; 1]	0.989180734963365205
6	$e^{x}+\sqrt{1+e^{2x}}=2$	[−1 ; 0]	–0.28768207245178092
7	$x\cdot\operatorname{tg}x=\frac{1}{3}$	[0.2 ; 1]	0.547160757260330037
8	$\sqrt{1-x}=\cos\sqrt{1-x}$	[0 ; 1]	0.453753165860328248

ВАРИАНТЫ ЗАДАНИЙ

Вариант 1

Полицейская машина

Стоящий на тротуаре пешеход наблюдает вращающуюся "мигалку" полицейской машины, движущейся по улице со скоростью v = 5 м/с. В начальный момент времени t = 0 расстояние между пешеходом и полицейской машиной минимально и составляет L = 10 м. В какие моменты времени пешеход будет видеть вспышки "мигалки", если едущий точно позади полицейской машины водитель такси наблюдает их при t = 0.125 с, 0.625 с, 1.125 с,... 5.125 с? Какое расстояние от точки наибольшего сближения с пешеходом пройдет в эти моменты полицейская машина? Каким при этом будет угол α (в градусах, см. рисунок)? Известно, что пешеход стоит справа по ходу машины, а луч "мигалки" вращается по часовой стрелке (если смотреть сверху).



Подсказки:

Перейдите в систему отсчета, связанную с полицейской машиной, и проведите ось X от начального положения пешехода в направлении против движения полицейской машины.

Как будет меняться координата пешехода $x_п$ со временем?

Каков период вращения "мигалки" T ? Как будет меняться со временем угол α (в радианах!) ? Как будет меняться со временем положение $x_{луча}$ точки пересечения луча "мигалки" с осью X?

Пешеход увидит вспышку в те моменты $t_{всп}$, когда выполняется условие $x_п(t_{всп}) = x_{луча}(t_{всп})$. Данное уравнение решите методом дихотомии с точностью 0.0001, используя в качестве начальных приближений времени значения $t_0 = nT - T/8$, $t_1 = nT + T/4$, где $n = 0, 1 \ldots 10$ – номер увиденной вспышки.

Результаты выведите в текстовый файл с именем FLASH.DAT, представив в виде таблицы:

n	Time	X	Alpha
0	0.000	0.00	0.00
1	0.520	2.60	14.58
2	1.038	5.19	27.43
3	1.553	7.76	37.82
4	2.064	10.32	45.90
5	2.572	12.86	52.14
6	3.079	15.40	56.99
7	3.584	17.92	60.84

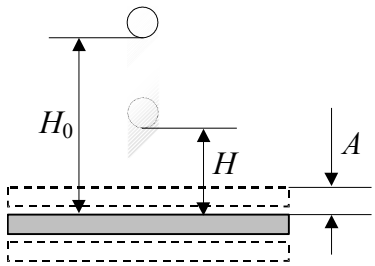
8	4.089	20.44	63.94
9	4.592	22.96	66.47
10	5.095	25.48	68.57

Пользуясь программой EASYPLOT, постройте график зависимости $\alpha(n)$.

Вариант 2

Вибростенд

Маленький стальной шарик в начальный момент времени падает из состояния покоя на массивную стальную горизонтальную плиту, совершающую гармонические колебания в вертикальном направлении по закону:



$$y_{пл} = A \cdot \sin\frac{2\pi t}{T} \quad .$$

На какую высоту H подскочит шарик после абсолютно упругого соударения с плитой, если он сброшен с высоты H_0? Решение проведите для H_0, меняющегося от 0.5 см до 100 см с шагом 0.5 см, A = 1 см, T = 0.1 с.

Подсказки:

В момент удара $t_{уд}$ шарик и плита имеют одинаковые координаты $y_{ш}(t_{уд}) = y_{пл}(t_{уд})$, причем $y_{ш}(t) = H_0 - \frac{gt^2}{2}$, где g = 981 см/с^2. Таким образом, для нахождения момента удара нужно решить уравнение

$$A \cdot \sin\frac{2\pi t_{уд}}{T} = H_0 - \frac{gt_{уд}^2}{2} .$$

Решите приведенное выше уравнение методом дихотомии с точностью 0.00001 с. Для нахождения границ корня t_0 и t_1 отметим, что удар о плиту произойдет наверняка при положительном значении времени, но раньше, чем свободно падающий шарик пройдет точку с координатой, равной $-A$. Отсюда получаем

$$t_0 = 0, \quad t_1 = \sqrt{\frac{2(H_0 + A)}{g}} \quad .$$

После удара скорость шарика относительно плиты меняет знак: $v'_{ш} - v_{пл} = -(v_{ш} - v_{пл})$, откуда

$$v'_{ш} = -v_{ш} + 2v_{пл},$$

причем

$$v_{ш} = -gt_{уд} \quad , \qquad v_{пл} = \frac{2\pi A}{T} \cdot \cos\frac{2\pi t_{уд}}{T} \quad .$$

Высота подскока шарика определяется как его координатой в момент удара, так и его скоростью после удара:

$$H = y_{ш}(t_{уд}) + \frac{(v'_{ш})^2}{2g} \quad .$$

Результаты решения задачи выведите в текстовый файл с именем PLATE.DAT в виде следующей таблицы:

H0	Time	V0	V`	H
0.5	0.008	31.3	118.5	7.63
1.0	0.017	44.3	79.4	4.08
1.5	0.071	54.2	37.8	-0.24
2.0	0.078	62.6	99.9	4.11
.....				
98.0	0.446	438.5	315.4	50.92
98.5	0.448	439.6	314.8	50.65
99.0	0.449	440.7	315.1	50.67
99.5	0.450	441.8	316.3	50.96
100.0	0.452	442.9	318.3	51.53

В этой таблице третья колонка – скорость шарика после удара от неподвижной плиты, вычисляемая по формуле $v_0 = \sqrt{2gH_0}$, а четвертая – скорость после удара от колеблющейся плиты $v'_{ш}$.

Пользуясь программой EASYPLOT, постройте графики зависимостей $H(H_0)$ и $v'_{ш}(H_0)$. Для наглядности на первом графике изобразите также прямую $H = H_0$ (высота подскока при соударении с неподвижной плитой), а на втором – кривую $v_0(H_0)$ (скорость после удара от неподвижной плиты).

Вариант 3

Проведите решение задачи предыдущего варианта, пользуясь для определения момента соударения шарика с плитой методом секущих вместо метода дихотомии. В качестве начальных для поиска корней используйте значения

$$t_0 = \sqrt{\frac{2H_0}{g}}, \quad t_1 = \sqrt{\frac{2(H_0 + A)}{g}} \quad .$$

Точность вычисления 0.00001 с.

Вариант 4

Лампочка

Рассчитайте зависимость температуры T нити лампочки для карманного фонарика, ее сопротивления R и силы тока в ней I от напряжения U, подаваемого на лампочку. Напряжение изменяйте от 0 до 4 В с шагом 0.1 В. Сопротивление нити при комнатной температуре $T_к$ = 300 К равно $R_к$ = 2 Ом. Считайте, что вся подводимая к лампочке электрическая мощность рассеивается в виде излучения в соответствии с законом Стефана-Больцмана

$$P_{изл} = S\sigma\left(T^4 - T_к^4\right) \quad , \text{ где}$$

$\sigma = 5.67 \cdot 10^{-8}$ Вт/(м^2·К^4) – постоянная Стефана-Больцмана, $S = 5 \cdot 10^{-7}$ м^2 – площадь излучающей поверхности нити лампочки. Сопротивление нити положите пропорциональным ее абсолютной температуре: $R = R_к \cdot (T/T_к)$.

Подсказки

Подводимая к лампочке электрическая мощность равна U^2/R, поэтому для нахождения температуры нити нужно при заданном значении напряжения на лампочке решить относительно T уравнение, выражающее закон сохранения энергии:

$$\frac{U^2 T_к}{R_к T} = S\sigma\left(T^4 - T_к^4\right) \quad .$$

Поиск корня проведите методом дихотомии в промежутке от 299 К до 3000 К с точностью 0.1 К.

Силу тока при заданном напряжении рассчитайте по закону Ома для участка цепи, используя для вычисления сопротивления полученное значение температуры нити:

$$I = \frac{U T_к}{R_к T} \quad .$$

Результаты решения задачи выведите в текстовый файл с именем TLAMP.DAT в виде следующей таблицы:

```
  U         T         R         I

 0.0       300      2.00     0.000
 0.1       565      3.77     0.027
 0.2       737      4.91     0.041
 0.3       865      5.76     0.052
.....................................

 3.5      2303     15.36     0.228
 3.6      2329     15.53     0.232
 3.7      2355     15.70     0.236
 3.8      2380     15.87     0.239
 3.9      2405     16.04     0.243
 4.0      2430     16.20     0.247
```

Пользуясь программой EASYPLOT, постройте графики зависимостей $T(U)$ и $I(U)$.

Вариант 5

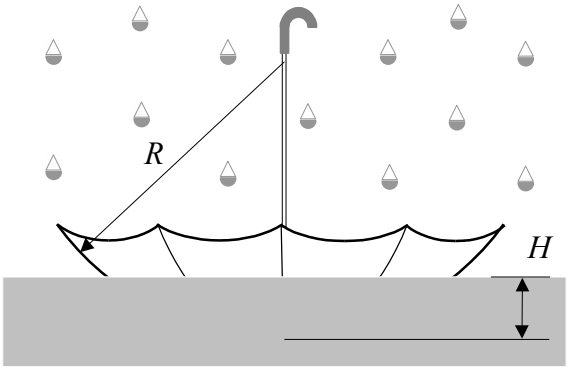
Проведите решение задачи предыдущего варианта, пользуясь методом секущих. В качестве начальных для поиска корней используйте значения 300 К и 400 К. Точность вычисления корня 0.1 К.

Вариант 6

Ковчег Винни-Пуха

Мы поплывем в твоем зонтике,– сказал Пух.
А.А.Милн. "Винни-Пух и все-все-все"

Во время наводнения, вызванного не прекращающимся много часов дождем с интенсивностью I = 2 см/час, Винни-Пух отправился спасать Пятачка, используя в качестве плавательного средства перевернутый зонтик. Из-за того, что зонтик постепенно наполнялся дождевой водой, глубина погружения "судна" H со временем увеличивалась, поскольку Пух не догадывался вычерпывать воду.



Рассчитайте зависимость глубины погружения зонтика с сидящим в нем Винни-Пухом от времени в интервале $0 \le t \le 4$ часов с шагом $\Delta t = 0.1$ часа, считая, что зонтик представляет собой сферический сегмент с радиусом R = 80 см. Площадь водосборной поверхности зонтика S = 8000 см^2, плотность воды ρ = 1 г/см^3, масса зонтика с сидящим в нем Винни-Пухом m_0 = 2000 г.

Подсказки

Интенсивность дождя I показывает, какой высоты слой воды образуется за единицу времени в открытом сосуде с вертикальными стенками. Значит, масса дождевой воды, собираемая на площади S за время t, равна ρISt.

Погруженная под воду часть зонтика имеет форму сферического сегмента высотой H. Объем сферического сегмента вычисляется по формуле

$$V = \frac{1}{3}\pi H^2 (3R - H) \quad .$$

В соответствии с законом Архимеда, вес зонтика с Винни-Пухом и набравшейся туда водой равен весу воды, вытесненной зонтиком. Следовательно, для нахождения глубины погружения зонтика необходимо для каждого указанного в условии значения t решить относительно H уравнение:

$$\rho ISt + m_0 = \frac{1}{3}\pi H^2 (3R - H) \cdot \rho \quad .$$

Поиск корня проведите методом дихотомии в промежутке от 0 см до 20 см с точностью 0.001 см.

Результаты решения задачи выведите в текстовый файл с именем VINNY.DAT в виде следующей таблицы:

Time, h	H, cm	M, kg
0.0	2.84	0.0
0.1	3.82	1.6
0.2	4.59	3.2
0.3	5.26	4.8
0.4	5.85	6.4
0.5	6.39	8.0
.....	.....	.....
3.5	15.72	56.0
3.6	15.94	57.6
3.7	16.16	59.2
3.8	16.38	60.8
3.9	16.59	62.4
4.0	16.80	64.0

В этой таблице последняя колонка представляет собой массу воды, собравшейся в зонтике (выраженная в килограммах).

Пользуясь программой EASYPLOT, постройте график зависимости $H(t)$.

Вариант 7

Проведите решение задачи предыдущего варианта, пользуясь методом секущих. В качестве начальных для поиска корней используйте значения $H_0 = 0$ см и $H_1 = 20$ см. Точность вычисления корня 0.001 см.

*Вариант 8**

Судьба детских воздушных шаров

Если срываются с ниток шары
То ли от дикой июльской жары,
То ли от качества ниток плохого,
То ли от ввысь устремленья лихого...

Борис Слуцкий

Сорвавшийся с нитки детский воздушный шарик, наполненный водородом, поднимается вверх. Рассчитайте, как изменяется действующая на него подъемная сила $F_{под}$ по

мере увеличения высоты h, если атмосферное давление изменяется с высотой в соответствии с барометрической формулой:

$$p_a = p_0 \cdot e^{-\frac{\mu g h}{RT}} ,$$

где $p_0 = 10^5$ Па – давление на уровне моря, $\mu = 0.029$ кг/моль – молярная масса воздуха, $g = 9.81$ м/с^2 – ускорение свободного падения, $R = 8.31$ Дж/(моль·К) – универсальная газовая постоянная, $T = 273$ К – абсолютная температура (предполагается одинаковой на всех высотах).

Масса водорода в шарике $m_{\text{Н}} = 1$ г, масса резиновой оболочки $m_0 = 5$ г, молярная масса водорода $\mu_{\text{Н}} = 0.002$ кг/моль. Считайте, что упругая сферическая оболочка шарика растягивается в соответствии с законом Гука, т.е. ее коэффициент поверхностного натяжения σ увеличивается при раздувании шарика по линейному закону:

$$\sigma = k \cdot (r - r_0) \text{ , где}$$

$r_0 = 0.1$ м – радиус нерастянутой оболочки, $k = 2 \cdot 10^4$ Па – коэффициент пропорциональности, имеющий смысл избыточного давления, требуемого для надувания шарика до радиуса $2r_0$.

Вычисления проведите для высот $0 \le h \le 20000$ м с шагом $\Delta h = 500$ м.

Подсказки

Подъемная сила показывает, насколько сила Архимеда превышает силу тяжести, действующую на шарик:

$$F_{под} = F_A - (m_H + m_0)g .$$

Силу Архимеда $F_A = \rho g V$ можно записать, выразив плотность воздуха ρ через атмосферное давление с помощью уравнения Клапейрона-Менделеева, а объем шарика V – через его радиус r:

$$F_A = \frac{p_a \mu}{RT} \cdot g \cdot \frac{4}{3} \pi r^3 .$$

Таким образом, для нахождения подъемной силы нужно атмосферное давление и радиус шарика подставить в рабочую формулу

$$F_{под} = \left(\frac{4\pi\mu}{3RT} \cdot p_a \cdot r^3 - m_H - m_0 \right) \cdot g \quad .$$

С одной стороны, по мере подъема шарика уменьшается атмосферное давление, т.е. плотность вытесняемого воздуха, но, с другой стороны, уменьшение давления вызывает увеличение радиуса шарика, т.е. объема вытесняемого воздуха. Конкуренция этих двух факторов приводит к сложной зависимости подъемной силы от высоты.

Для нахождения радиуса шарика учтем, что на находящийся в нем газ действует давление атмосферы плюс лапласово давление со стороны стремящейся сократиться упругой оболочки: $p = p_a + 2\sigma / r$. С другой стороны, давление данной массы газа в шарике связано с его объемом уравнением Клапейрона-Менделеева: $p = m_H RT / \mu_H V$. Выразив объем шарика и значение σ через радиус, получаем следующее нелинейное уравнение относительно r:

$$\frac{3m_H RT}{4\pi\mu_H \cdot r^3} = \frac{2k(r - r_0)}{r} + p_a \quad .$$

Поиск корня проведите методом секущих. В качестве первого пробного значения корня используйте радиус r_0, в качестве второго – радиус, который имел бы шарик, если бы его расширение не ограничивалось упругостью резиновой оболочки (т.е. $k = 0$)

$$r_1 = \sqrt[3]{\frac{3m_H RT}{4\pi\mu_H \cdot p_a}} \quad .$$

Решение проведите с точностью 0.0001 м.

Таким образом, для решения задачи необходимо для каждого значения высоты (т.е. в цикле по h)

– вычислить давление по барометрической формуле;
– найти пробные значения радиусов r_0 и r_1;
– методом секущих найти радиус пузыря;
– подставить атмосферное давление и радиус в формулу для вычисления подъемной силы;
– вывести полученные результаты.

Результаты решения задачи выведите в текстовый файл с именем AIRBALL.DAT в виде таблицы, содержащей значения высоты в километрах, атмосферного давления в долях от p_0, радиуса шарика в сантиметрах, подъемной силы в миллиньютонах:

h,km	p/p0	r,cm	F,mN
0.0	1.000	13.49	70.05
0.5	0.939	13.72	68.65
1.0	0.882	13.96	67.18
1.5	0.829	14.20	65.62
2.0	0.778	14.43	63.99
.....			
18.0	0.105	20.59	-10.90
18.5	0.098	20.70	-13.04
19.0	0.092	20.82	-15.12
19.5	0.087	20.93	-17.13
20.0	0.081	21.03	-19.09

Пользуясь программой EASYPLOT, постройте графики зависимости $r(h)$ и $F_{под}(h)$.

* * *

Глава 4

Решение задач, приводящих к системам линейных алгебраических уравнений

1. *Краткая теория*

Во многих задачах физики, например, в теории колебаний, при расчете разветвленных электрических цепей, в теории твердого тела и др. приходится сталкиваться с системами линейных алгебраических уравнений.

Системой n линейных алгебраических уравнений с n неизвестными называется система вида

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases} \quad (1)$$

где $x_1, x_2, \ldots x_n$ - неизвестные. Свойства системы зависят от ее *определителя D* – числа, которое находится с помощью некоторого правила по коэффициентам системы. Определитель обозначается

$$D = \begin{vmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ & \ldots & \ldots & \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{vmatrix}$$

Доказано, что если определитель системы не равен нулю, то система (1) имеет единственное решение. Это значит, что существует такой единственный набор чисел

$x_1, x_2, \ldots x_n$, который обращает все уравнения системы в равенства.

Решение системы линейных уравнений, в частности, может быть найдено с помощью правила Крамера:

$$x_k = \frac{D_k}{D} \quad (k = 1, 2, \ldots n),$$

где D_k - определитель, получающийся из D при замене элементов k-го столбца $a_{1k}, a_{2k}, \ldots a_{nk}$ соответствующими свободными членами $b_1, b_2, \ldots b_n$. Однако формула Крамера удобна для теоретического исследования свойств решения, но весьма невыгодна для его численного нахождения. Дело в том, что для непосредственного вычисления определителей системы с n неизвестными требуется порядка $n!n$ арифметических операций ($n!$ = $1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$), и уже при n = 30 такое число операций недоступно современным компьютерам.

Одним из наиболее известных методов решения систем линейных алгебраических уравнений является *метод исключения Гаусса.* Прежде, чем рассматривать этот метод, вспомним два важных свойства систем линейных уравнений.

- Решение системы не изменится, если правую и левую часть каждого уравнения умножить на любое число, отличное от нуля.
- Решение системы не изменится, если к любому уравнению системы почленно прибавить (или вычесть) другое уравнение этой системы.

Метод исключения Гаусса основан на приведении системы к так называемому треугольному виду на основе использования упомянутых свойств. Рассмотрим его на примере следующей системы:

$$\begin{cases} 4x_1 + 3x_2 + x_3 = 13 \\ 2x_1 - x_2 - x_3 = -3 \\ 7x_1 + x_2 - 3x_3 = 0 \end{cases} \qquad (2)$$

Вычтем из второго уравнения системы первое, умноженное на такое число, чтобы уничтожился коэффициент при x_1 (это число 1/2). Точно так же добьемся исчезновения

первого слагаемого в третьем уравнении, вычитая из него первое уравнение, умноженное на 7/4. В результате получим:

$$\begin{cases} 4x_1 + 3x_2 + x_3 = 13 \\ -\frac{5}{2}x_2 - \frac{3}{2}x_3 = -\frac{19}{2} \\ -\frac{17}{4}x_2 - \frac{19}{4}x_3 = -\frac{91}{4} \end{cases}$$

Аналогично в третьем уравнении избавимся от слагаемого с x_2, вычитая из него второе уравнение, умноженное на -17/10.

$$\begin{cases} 4x_1 + 3x_2 + x_3 = 13 \\ -\frac{5}{2}x_2 - \frac{3}{2}x_3 = -\frac{19}{2} \\ -\frac{11}{5}x_3 = -\frac{33}{5} \end{cases} \quad (3)$$

Мы получили систему, приведенную к треугольному виду, решение которой совпадает с решением исходной системы (2). На этом завершается первая часть метода Гаусса, которая называется *прямым ходом*.

Вторая часть метода Гаусса – *обратный ход* – состоит в последовательном нахождении неизвестных, начиная с последнего уравнения системы (3). Так, в нашем случае из третьего уравнения непосредственно получаем $x_3 = 3$, подставляя это значение во второе уравнение, находим $x_2 = 2$, и, наконец, зная эти два неизвестных, из первого уравнения имеем $x_1 = 1$.

2. Реализация метода исключения Гаусса в библиотечном модуле Numerics

Метод исключения Гаусса используется в процедуре с именем `LinSys`. Эта процедура не входит в состав стандартных процедур Turbo-Pascal, таких, как, например, `Writeln` или `Assign`. Она включена в библиотеку численных методов, написанную автором специально для учебных и научных целей и организованную в виде так называемого *модуля*. Этот модуль имеет имя `Numerics`.

Модуль очень похож на обычную Pascal-программу, но его нельзя непосредственно выполнить. Модуль представляет собой хранилище процедур, функций и других объектов, которые готовы к использованию в других программах. При этом нет необходимости знать подробности работы этих процедур и функций. Достаточно указать, что данная программа использует модуль, где содержится их откомпилированный машинный код. Это делается с помощью выражения

```
uses <имя модуля>;
```

которое помещается в самом начале раздела описаний программы.

Например, для доступа к процедуре `LinSys` из модуля `Numerics` программа перед описаниями всех констант, переменных и т.д. должна содержать строку

```
uses Numerics;
```

Модуль `Numerics` в виде готового машинного кода содержится в файле NUMERICS.TPU, который должен находиться на диске компьютера либо в текущем каталоге, либо в каталоге, указанном при настройке среды Turbo-Pascal. Исходный текст модуля находится в файле NUMERICS.PAS, распечатка которого дана в приложении.

3. Как пользоваться процедурой LinSys

Перед решением системы с помощью процедуры `LinSys` необходимо составить специальную таблицу из коэффициентов системы, которая называется *расширенной матрицей*. Левый столбец этой матрицы (будем считать его нулевым) составляется из свободных членов $b_1, b_2, \ldots b_n$, а столбцы с 1-го по *n*-й образованы коэффициентами при неизвестных:[1]

$$\begin{pmatrix} b_1 & a_{11} & a_{12} & \ldots & a_{1n} \\ b_2 & a_{21} & a_{22} & \ldots & a_{2n} \\ & \ldots. & \ldots. & \ldots. & \\ b_n & a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix}$$

[1] В математической литературе столбец свободных членов обычно записывается справа.

Например, для системы (2) расширенная матрица записывается так:

$$\begin{pmatrix} 13 & 4 & 3 & 1 \\ -3 & 2 & -1 & -1 \\ 0 & 7 & 1 & -3 \end{pmatrix}$$

Обращение к процедуре LinSys имеет вид:

```
LinSys (N, Matr, A, Det);
```

Здесь

N - число уравнений системы, integer;

Matr - расширенная матрица коэффициентов системы. Элементы матрицы должны иметь тип extended.

A - массив из N элементов типа extended. Сюда процедура LinSys записывает вычисленное решение системы;

Det - переменная типа extended. Сюда процедура записывает вычисленный определитель системы. В случае, если определитель системы равен нулю, процедура выводит на экран сообщение:

```
НУЛЕВОЙ ОПРЕДЕЛИТЕЛЬ СИСТЕМЫ
  ЕДИНСТВЕННОГО РЕШЕНИЯ НЕТ
```

и выполнение программы прекращается.

Ниже приводится программа Lin1 решения системы (2) с использованием процедуры LinSys, а также результаты ее работы. Программа начинается с директив компилятору, обеспечивающих подключение математического сопроцессора для обработки чисел в формате extended или его программную эмуляцию. Далее указывается, что программа использует библиотечный модуль Numerics. В разделе описания констант задается число уравнений NEqu и константа-массив A, содержащая расширенную матрицу системы размером 3 строки на 4 столбца. В разделе описания переменных описывается массив X из трех элементов для записи решения системы и переменная D для записи вычисленного определителя. Целочисленная переменная i используется в дальнейшем в качестве счетчика при выводе значений корней.

Основная часть программы предельно проста. Поскольку все исходные данные уже заданы в разделе описания констант, она сразу начинается с обращения к процедуре

решения системы, после чего следует не требующий пояснений цикл вывода значений неизвестных и вывод значения определителя системы.

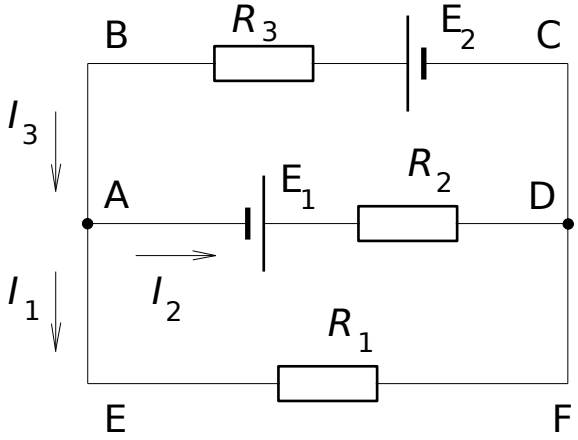
Отметим, что чаще всего для решения задачи не требуется вычисления определителя, однако в любом случае для работы процедуры LinSys под него нужно выделять переменную в разделе описаний.

```
program Lin1;

      {$N+,E+}
uses Numerics;

const
  NEqu = 3;  { число уравнений системы }

  A: array [1..NEqu, 0..NEqu] of extended =
               { расширенная матрица }
            ( ( 13,    4,     3,     1 ),
              ( -3,    2,    -1,    -1 ),
              (  0,    7,     1,    -3 ) ) ;

var
  X: array [1..NEqu] of extended;
  D: extended;
  i: integer;

BEGIN

  LinSys (NEqu, A, X, D);

      { вывод корней и определителя }
  for i:= 1 to NEqu do
    Writeln ('X[', i, '] =', X[i]:7:4);
  Writeln ('Определитель равен', D:8:4)

END.

X[1] = 1.0000
X[2] = 2.0000
X[3] = 3.0000
Определитель равен 22.0000
```

4. Компьютерный расчет разветвленных электрических цепей

Следующий пример демонстрирует использование процедуры LinSys для расчета разветвленных

электрических цепей. Пусть имеется электрическая цепь из источников ЭДС и резисторов, изображенная на рисунке. Требуется рассчитать значения токов через каждый резистор.



R_1 = 30 Ом, R_2 = 20 Ом, R_3 = 40 Ом,
E_1 = 80 В, E_2 = 45 В.

Расчет сводится к решению системы линейных уравнений, которые составляются на основании *правил Кирхгофа* и содержат в качестве неизвестных значения токов.

Согласно правилам Кирхгофа:

- для любого узла цепи сумма токов втекающих в узел, равна сумме токов, вытекающих из узла (правило узлов);
- алгебраическая сумма падений напряжения на всех элементах любого замкнутого контура равна алгебраической сумме всех ЭДС, действующих в этом контуре (правило контуров).

Вначале стрелками отметим направления токов, которые мы принимаем за положительные. Пусть I_1 - ток, протекающий через резистор R_1, и т.д. Первое правило Кирхгофа применим для узла A, а второе – для контуров ABCD и ADEF. В результате получаем три следующих уравнения:

$$\begin{cases} I_1 + I_2 - I_3 = 0 \\ R_2 I_2 + R_3 I_3 = E_1 + E_2 \\ -R_1 I_1 + R_2 I_2 = E_1 \end{cases}$$

Расширенная матрица системы имеет вид:

$$\begin{pmatrix} 0 & 1 & 1 & -1 \\ E_1 + E_2 & 0 & R_2 & R_3 \\ E_1 & -R_1 & R_2 & 0 \end{pmatrix}$$

Ниже приводится программа, решающая систему линейных уравнений с данной матрицей, и результаты ее работы. Программа не нуждается в особых комментариях. Обратим внимание лишь на то, что начиная с версии 6.0 Turbo-Pascal допускает использование арифметических выражений при описании констант (это использовано при описании матрицы A). Отметим также, что поскольку значения ЭДС заданы в вольтах, а сопротивлений - в омах, решение системы представляет собой токи, выраженные в амперах.

```
program Lin2;

      {$N+,E+}

uses Numerics;

const
  NEqu = 3;  { число уравнений системы }

      { Значения сопротивлений и ЭДС }
  R1 = 30; R2 = 20; R3 = 40;
  E1 = 80; E2 = 45;

  A: array [1..NEqu, 0..NEqu] of extended =
               { расширенная матрица }
            ( (   0,     1,     1,     -1 ),
              ( E1+E2,   0,     R2,    R3 ),
              (   E1,   -R1,    R2,     0 ) ) ;
var
  I: array [1..NEqu] of extended;
  D: extended;
  k: integer;
```

```
BEGIN

  LinSys (NEqu, A, I, D);

      { вывод токов через резисторы }
  for k:= 1 to NEqu do
    Writeln ('I', k,' =', I[k]:7:4, ' ампер');

  Readln
END.
```

```
I1 =-0.8846 ампер
I2 = 2.6731 ампер
I3 = 1.7885 ампер
```

Очевидно, отрицательное значение тока через первый резистор означает, что ток через него протекает в направлении, противоположном указанному стрелкой на схеме.

ТРЕНИРОВОЧНЫЕ ПРИМЕРЫ

Напишите и отладьте программу решения предложенных ниже систем трех линейных алгебраических уравнений с использованием процедуры `LinSys`. Результаты выведите на экран в таком же виде, как в ответах к вариантам заданий.

Вариант	Система	Решение

1	$\begin{cases} 3x_1 - 1.5x_2 + 0.5x_3 = 0.9 \\ 1.5x_1 + 2.5x_2 - x_3 = 1.5 \\ x_1 + 4x_2 - 1.5x_3 = 2 \end{cases}$	`X[1] = 0.5600`
		`X[2] = 1.8000`
		`X[3] = 3.8400`

2	$\begin{cases} 1.5x_1 - 0.5x_2 + 0.2x_3 = 0.4 \\ -0.5x_1 + 1.5x_2 - 0.3x_3 = 0.8 \\ -0.3x_1 + 0.2x_2 - 0.5x_3 = 0.2 \end{cases}$	`X[1] = 0.5356`
		`X[2] = 0.6169`
		`X[3] =-0.4746`

3	$\begin{cases} 5x_1 - 5x_2 + 2x_3 = 1.9 \\ -5x_1 + 5x_2 - 3x_3 = 0.3 \\ -3x_1 + 2x_2 - 5x_3 = -1.8 \end{cases}$	X[1] =10.2800 X[2] = 9.0200 X[3] =-2.2000
4	$\begin{cases} 2x_1 - 6x_2 - 10x_3 = -2 \\ 5x_1 + x_2 + 20x_3 = 9 \\ -3x_1 + 2x_2 - 10x_3 = 5 \end{cases}$	X[1] = 6.8824 X[2] = 5.1765 X[3] =-1.5294
5	$\begin{cases} 2.5x_1 - 3x_2 - 1.2x_3 = -1 \\ -3.5x_1 + 2.6x_2 + 2.0x_3 = -1.5 \\ -6.5x_1 - 2x_2 - 10x_3 = -1.8 \end{cases}$	X[1] = 3.7305 X[2] = 2.3557 X[3] = 2.7160
6	$\begin{cases} 1.02x_1 - 0.25x_2 - 0.3x_3 = 0.515 \\ -0.41x_1 + 1.13x_2 - 0.15x_3 = 1.555 \\ -0.25x_1 - 0.14x_2 - 1.21x_3 = 2.780 \end{cases}$	X[1] = 2.0000 X[2] = 2.5000 X[3] = 3.0000
7	$\begin{cases} 6x_1 - x_2 - x_3 = 11 \\ -x_1 + 6x_2 - x_3 = 32 \\ -x_1 - x_2 + 6x_3 = 42 \end{cases}$	X[1] = 4.6071 X[2] = 7.6071 X[3] = 9.0357
8	$\begin{cases} 6.1x_1 + 2.2x_2 + 1.2x_3 = 16.55 \\ 2.2x_1 + 5.5x_2 - 1.5x_3 = 10.55 \\ 1.2x_1 - 1.5x_2 + 7.2x_3 = 16.80 \end{cases}$	X[1] = 1.5000 X[2] = 2.0000 X[3] = 2.5000

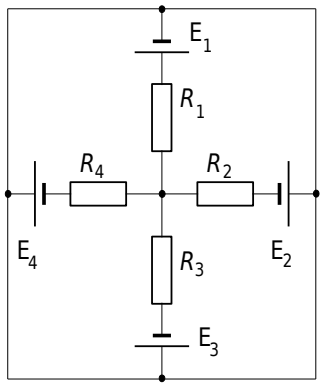
ЗАДАНИЕ

В соответствии с предложенным вариантом, рассчитайте значения токов через все резисторы изображенной на схеме электрической цепи, пользуясь процедурой `LinSys`. Результаты расчета выведите на экран в таком же виде, как в ответах к вариантам заданий.

В отчете по данной работе приведите:

– принципиальную схему рассчитываемой электрической цепи с указанием выбранных узлов и контуров;
– полученную из анализа схемы систему линейных уравнений;
– расширенную матрицу системы;
– программу расчетов;
– полученные результаты.

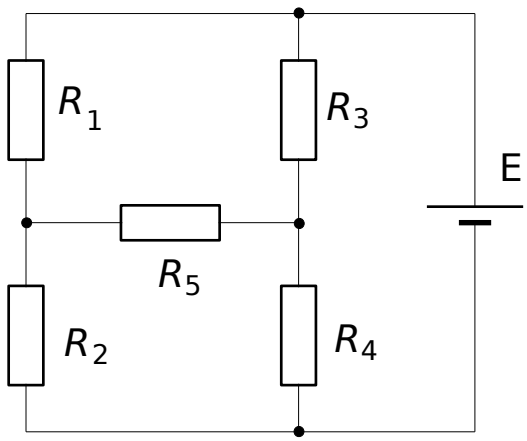
Вариант 1




```
Ответ:

I1 = 0.3920 ампер
I2 = 0.1040 ампер
I3 = 0.1360 ампер
I4 = 0.1520 ампер
```

E_1 = 1 В, E_2 = 2 В, E_3 = 3 В, E_4 = 4 В,
R_1 = 5 Ом, R_2 = 10 Ом, R_3 = 15 Ом, R_4 = 20 Ом.

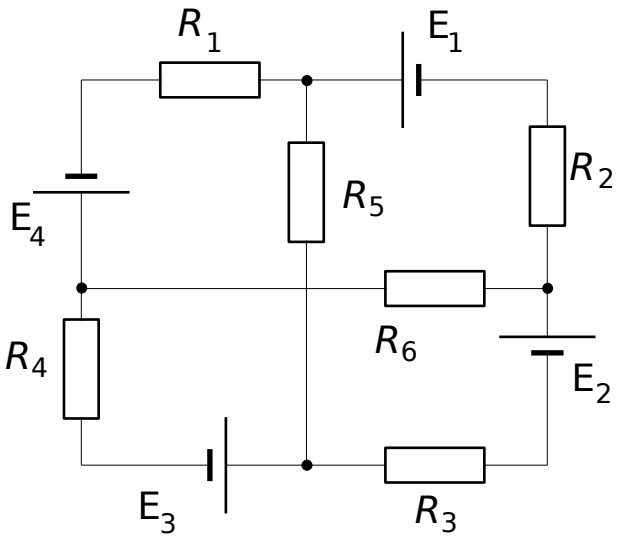
Вариант 2




```
Ответ:

I1 = 3.2877 ампер
I2 = 3.4247 ампер
I3 = 1.3014 ампер
I4 = 1.1644 ампер
I5 = 0.1370 ампер
```

E = 10 В,
R_1 = 2 Ом, R_2 = 1 Ом, R_3 = 5 Ом, R_4 = 3 Ом, R_5 = 0.5 Ом.

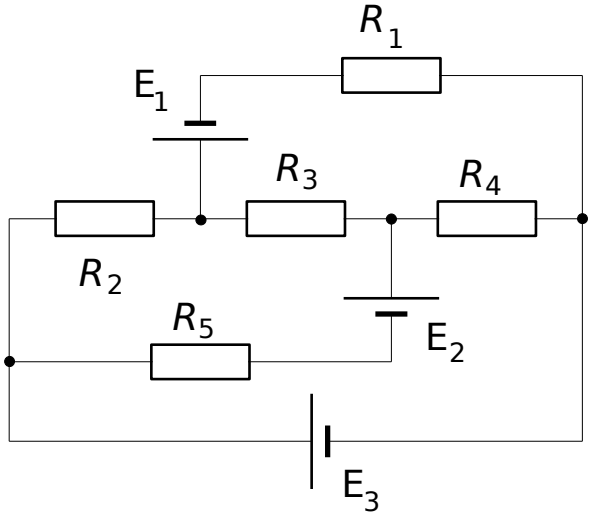
Вариант 3




```
Ответ:

I1 = 1.0879 ампер
I2 = 0.5824 ампер
I3 = 1.1319 ампер
I4 = 1.6374 ампер
I5 =-0.5055 ампер
I6 =-0.5495 ампер
```

E_1 = 1 В, E_2 = 2 В, E_3 = 3 В, E_4 = 4 В,
R_1 = 4 Ом, R_2 = 3 Ом, R_3 = 2 Ом, R_4 = 1 Ом, R_5 = 2 Ом,
R_6 = 2 Ом.

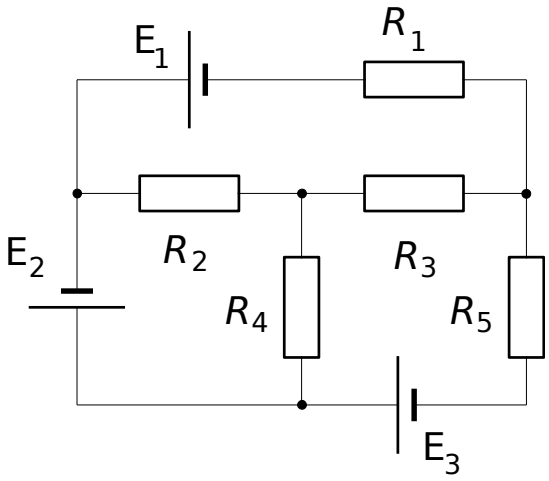
Вариант 4




```
Ответ:

I1 =-3.6478 ампер
I2 = 3.1761 ампер
I3 =-0.4717 ампер
I4 = 2.5157 ампер
I5 = 2.9874 ампер
```

E_1 = 10 В, E_2 = 20 В, E_3 = 30 В,
R_1 = 2 Ом, R_2 = 4 Ом, R_3 = 6 Ом, R_4 = 8 Ом, R_5 = 10 Ом.

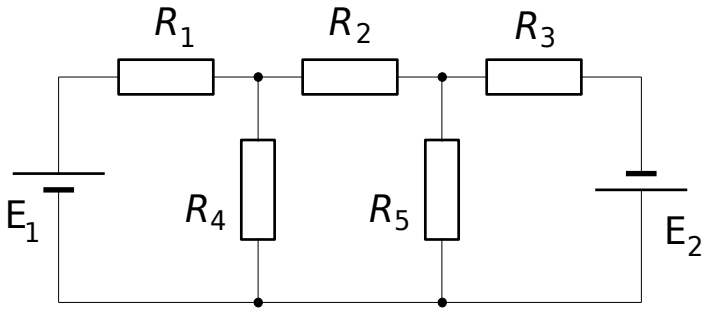
Вариант 5




```
Ответ:

I1 = 0.2688 ампер
I2 =-0.1183 ампер
I3 = 0.3226 ампер
I4 = 0.4409 ампер
I5 = 0.0538 ампер
```

E_1 = 1 В, E_2 = 2 В, E_3 = 3 В,
R_1 = 1 Ом, R_2 = 2 Ом, R_3 = 3 Ом, R_4 = 4 Ом, R_5 = 5 Ом.

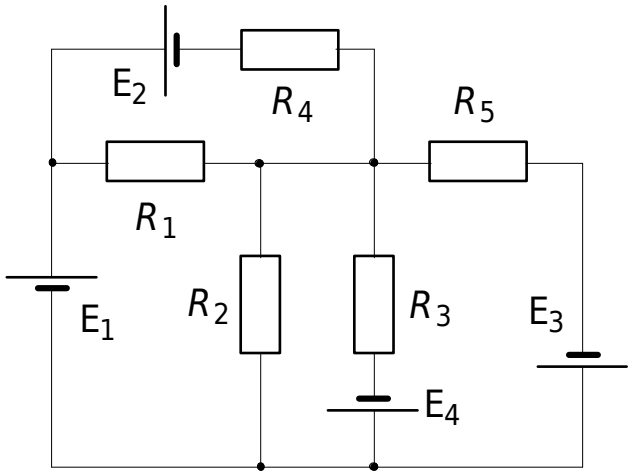
Вариант 6




```
Ответ:

I1 = 4.2857 ампер
I2 = 2.8571 ампер
I3 = 4.2857 ампер
I4 = 1.4286 ампер
I5 = 1.4286 ампер
```

E_1 = 10 В, E_2 = 10 В,
R_1 = 2 Ом, R_2 = 1 Ом, R_3 = 2 Ом, R_4 = 1 Ом, R_5 = 1 Ом.

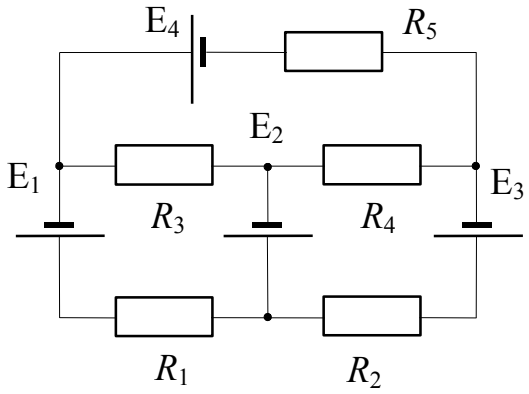
Вариант 7




```
Ответ:

I1 = 3.8611 ампер
I2 = 1.1389 ампер
I3 = 1.0556 ампер
I4 =-2.4259 ампер
I5 = 4.0926 ампер
```

E_1 = 10 В, E_2 = 5 В, E_3 = 5 В, E_4 = 3 В,
R_1 = 2 Ом, R_2 = 2 Ом, R_3 = 5 Ом, R_4 = 3 Ом, R_5 = 3 Ом.

Вариант 8




```
Ответ:

I1 =-0.3182 ампер
I2 =-1.1364 ампер
I3 = 1.1364 ампер
I4 = 0.3182 ампер
I5 = 0.8182 ампер
```

E_1 = 5 В, E_2 = 10 В, E_3 = 5 В, E_4 = 10 В,
R_1 = 5 Ом, R_2 = 3 Ом, R_3 = 3 Ом, R_4 = 5 Ом, R_5 = 10 Ом.

* * *

Глава 5

Обработка результатов эксперимента методом наименьших квадратов

1. Понятие о методе наименьших квадратов

Пусть некоторое физическое явление характеризуется двумя величинами, одна из которых является независимой (величина *x*), а другая зависимой от первой (величина *y*). Предположим, что связь между этими величинами исследуется экспериментально, и в результате измерений ряду значений $x\,(x_1, x_2, \ldots x_m)$ поставлены в соответствие некоторые значения величины $y\,(y_1, y_2, \ldots y_m)$, то есть получены *m* экспериментальных точек. Естественно, что из-за погрешностей эксперимента значения y_i определены с ошибками.

Требуется подобрать *аналитическую* (в виде формулы) зависимость $y = f(x)$, связывающую переменные *x* и *y*. Аналитические зависимости, подобранные на основании экспериментальных наблюдений, называются *эмпирическими*. Выявление эмпирических зависимостей делится на два этапа – выбор эмпирической формулы и определение величин параметров (*эмпирических коэффициентов*), входящих в эту формулу. Первая задача обычно решается на основании теоретического исследования наблюдаемого явления. Методом наименьших квадратов решается вторая задача.

Пусть искомая эмпирическая зависимость $y = f(x)$ имеет вид

$$y = a_1\varphi_1(x) + a_2\varphi_2(x) + \ldots + a_n\varphi_n(x)$$

(1),

где функции $\varphi_1(x), \varphi_2(x), \ldots \varphi_n(x)$ заданы, а стоящие перед ними коэффициенты $a_1, a_2, \ldots a_n$ подлежат определению. Например,

$y = a_1 + a_2x + a_3x^2 + a_4x^3$ (здесь n = 4).

Так как значения y_i определены из эксперимента с ошибками, то для сглаживания этих ошибок число экспериментальных точек должно существенно превышать число эмпирических коэффициентов ($m > n$).

В рассматриваемом здесь варианте метода наименьших квадратов эмпирические коэффициенты a_j обязательно должны входить в эмпирическую формулу линейно, т.е. в соответствии с формулой (1). Если это условие не выполняется, нужно попытаться найти такую замену переменных, чтобы в новых переменных неизвестные коэффициенты линейно входили в формулу. Например, зависимость $p = \alpha \cdot \exp \beta x$ логарифмированием можно представить в виде $y = a_1 + a_2x$, где $y = \ln p$, $a_1 = \ln \alpha$, $a_2 = \beta$.

Подставим в выражение (1) последовательно все m экспериментальных значений x_i и y_i.

$$\begin{cases} a_1\varphi_1(x_1) + a_2\varphi_2(x_1) + \ldots + a_n\varphi_n(x_1) = y_1 \\ a_1\varphi_1(x_2) + a_2\varphi_2(x_2) + \ldots + a_n\varphi_n(x_2) = y_2 \\ \quad \ldots\ldots\ldots\ldots\ldots \\ a_1\varphi_1(x_m) + a_2\varphi_2(x_m) + \ldots + a_n\varphi_n(x_m) = y_m \end{cases}$$

(2).

Мы получили систему m линейных уравнений с n неизвестными $a_1 \text{ --- } a_n$. Поскольку $m > n$ то данная система не имеет решений в обычном смысле этого слова, т.е. не существуют такие значения a_j, при которых превращались бы в равенства все уравнения системы.

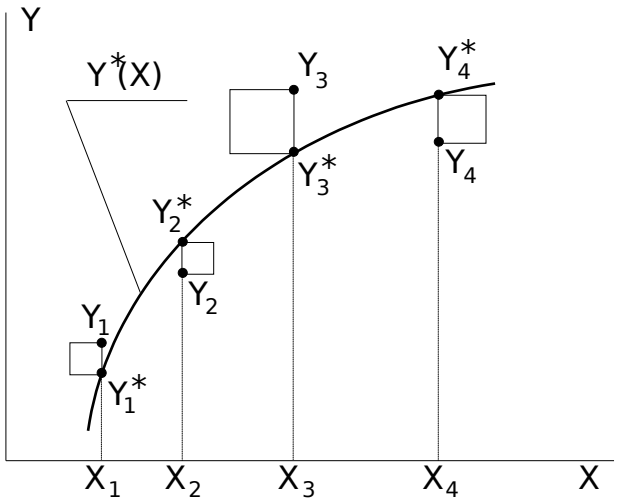
Однако решение системы (2) возможно в смысле метода наименьших квадратов, который состоит в следующем. Обозначим левые части уравнений системы через $y_1^*, y_2^*, \ldots y_m^*$:

$$y_i^* = \sum_{j=1}^{n} a_j \varphi_j(x_i) \qquad (3).$$

Отметим, что y_i^* – это не что иное, как значения физической величины *y*, полученные при *расчете* по эмпирической формуле (1). Разумеется, значения y_i^* не совпадают с *экспериментально полученными* значениями y_i. Однако можно найти такие эмпирические коэффициенты a_j, чтобы сумма квадратов отклонений расчетных значений *y* от экспериментальных была минимальной:

$$S = \sum_{i=1}^{m} \left(y_i - y_i^* \right)^2 = \min \qquad (4).$$

При этом коэффициенты a_j называются решениями системы линейных уравнений (2) методом наименьших квадратов.



С геометрической точки зрения решить систему (2) методом наименьших квадратов (применительно к обработке результатов эксперимента) означает найти такие значения эмпирических коэффициентов a_j, при которых суммарная площадь квадратов, построенных на отрезках – отклонениях экспериментальных результатов от расчетных, будет минимальной (см. рисунок).

В заключение необходимо еще раз отметить, что число уравнений системы (2) соответствует числу экспериментальных точек *m*, а число неизвестных равно числу *n* коэффициентов в эмпирической формуле.

2. Реализация метода наименьших квадратов в библиотечном модуле Numerics

Для решения системы (2) методом наименьших квадратов могут быть использованы стандартные процедуры, входящие в состав библиотечного модуля `Numerics`.

Теория метода наименьших квадратов, детальное рассмотрение которой не входит в наши задачи, позволяет найти эмпирические коэффициенты a_j путем преобразования системы уравнений (2) в *систему n линейных уравнений с n неизвестными* $a_1 \dots a_n$ вида:

$$\begin{cases} f_{11}a_1 + f_{12}a_2 + \dots + f_{1n}a_n = b_1 \\ f_{21}a_1 + f_{22}a_2 + \dots + f_{2n}a_n = b_2 \\ \quad \dots\dots\dots \\ f_{n1}a_1 + f_{n2}a_2 + \dots + f_{nn}a_n = b_n \end{cases} \quad (5).$$

Можно показать, что коэффициенты этой системы выражаются через коэффициенты исходной системы (2):

$$f_{ik} = \sum_{j=1}^{m} \varphi_i(x_j) \cdot \varphi_k(x_j); \quad b_i = \sum_{j=1}^{m} \varphi_i(x_j) \cdot y_j \qquad (6).$$

Таким образом, для нахождения эмпирических коэффициентов a_j необходимо:

1. преобразовать систему (2) в систему (5), пользуясь формулами (6).
2. решить систему (5).

Первый этап решения задачи осуществляет процедура `MinSqu`, входящая в библиотечный модуль `Numerics`, второй этап – решение системы (5) – осуществляет процедура `LinSys`, из этого же модуля. Использование процедуры `LinSys` было подробно описано в главе 4.

3. Примеры использования процедур MinSqu и LinSys

Обращение к процедуре MinSqu имеет вид:

```
MinSqu (M, N, Matr1, Matr2);
```

Здесь

M – число уравнений исходной системы (2) (т.е. число экспериментальных точек), integer;

N – число неизвестных (т.е. число эмпирических коэффициентов), integer;

Matr1 – расширенная матрица коэффициентов исходной системы (2). Она должна состоять из M строк с номерами от 1 до M и из N+1 столбца с номерами от 0 до N. Нулевой (левый) столбец – столбец свободных членов. Элементы матрицы должны иметь тип extended;

Matr2 – расширенная матрица системы (5) – результат работы процедуры MinSqu. Она состоит из N строк с номерами от 1 до N и N+1 столбцов с номерами от 0 до N. Нулевой (левый) столбец – столбец свободных членов. Элементы матрицы должны иметь тип extended.

Система (5) с расширенной матрицей Matr2 решается с помощью процедуры LinSys, обращение к которой имеет вид:

```
LinSys (N, Matr, A, Det);
```

Здесь

N – число уравнений системы, integer;

Matr – расширенная матрица коэффициентов системы (в нашем случае сюда подставляем Matr2). Элементы матрицы должны иметь тип extended.

A – массив из N элементов типа extended. Сюда процедура LinSys записывает решение системы;

Det – переменная типа extended. Сюда процедура записывает определитель системы.

Рассмотрим ряд примеров использования процедур MinSqu и LinSys для решения систем линейных алгебраических уравнений методом наименьших квадратов.

Пусть в результате какого-либо эксперимента получена следующая зависимость $y(x)$:

X	7	12	17	22	27	32	37
Y	83.7	72.9	63.2	54.7	47.5	41.4	36.3

Требуется описать эту зависимость эмпирической формулой вида

$$y = a_1 + a_2 x + a_3 x^2 \qquad (7)$$, где

a_1, a_2, a_3 – эмпирические коэффициенты, подлежащие определению. Подставляя экспериментальные значения x и y в формулу (7), получим систему семи уравнений с тремя неизвестными:

$$\begin{cases} a_1 + 7a_2 + 7^2 a_3 = 83.7 \\ a_1 + 12a_2 + 12^2 a_3 = 72.9 \\ \quad \cdots\cdots\cdots \\ a_1 + 37a_2 + 37^2 a_3 = 36.3 \end{cases} \qquad (8).$$

Обозначим расширенную матрицу этой системы C_{ij}:

$$\{C_{ij}\} = \begin{pmatrix} 83.7 & 1 & 7 & 49 \\ 72.9 & 1 & 12 & 144 \\ 63.2 & 1 & 17 & 289 \\ 54.7 & 1 & 22 & 484 \\ 47.5 & 1 & 27 & 729 \\ 41.4 & 1 & 32 & 1024 \\ 36.3 & 1 & 37 & 1369 \end{pmatrix} \qquad (9).$$

Программа `SYS1` обеспечивает решение системы, определенной матрицей (9).

Поскольку в программе используется тип `extended`, программа начинается с директив подключения математического сопроцессора и его программной эмуляции в случае, если сопроцессор отсутствует в данном компьютере. Далее описываются константы, определяющие число уравнений `M` и число неизвестных `N` в системе.

Расширенная матрица системы C также задается в виде константы - двумерного массива.

В разделе описания переменных описаны расширенная матрица Matr системы (5) для записи результатов работы процедуры MinSqu, массив A, куда записывается решение системы (5), т.е. искомые эмпирические коэффициенты, переменная Det для записи определителя системы и, наконец, целочисленная переменная i, которая служит счетчиком элементов массива A.

Раздел операторов программы предельно прост. Он состоит из последовательного вызова процедур MinSqu и LinSys, после чего следует цикл вывода на экран результатов решения системы.

```
program SYS1;
      {$N+,E+}
uses Numerics;

const
  M = 7;     { число уравнений = числу эксп. точек }
  N = 3;     { число неизвестных = числу эмпир. коэфф. }

  C: array [1..M, 0..N] of extended =
                { расширенная матрица коэффициентов }
            ( ( 83.7,  1,   7,   49 ),
              ( 72.9,  1,  12,  144 ),
              ( 63.2,  1,  17,  289 ),
              ( 54.7,  1,  22,  484 ),
              ( 47.5,  1,  27,  729 ),
              ( 41.4,  1,  32, 1024 ),
              ( 36.3,  1,  37, 1369 ) );
var
  Matr: array [1..N, 0..N] of extended;
  A: array [1..N] of extended;
  Det: extended;
  i: integer;

BEGIN
      { преобразование системы M*N в систему N*N }
  MinSqu (M, N, C, Matr);
      { решение системы N*N }
  LinSys (N, Matr, A, Det);
      { вывод эмпирических коэффициентов }
  for i:= 1 to N do
    Writeln ('A[', i ,'] =', A[i]:10:5)
END.
```

Ниже приведены результаты работы программы SYS1.

```
A[1] = 100.79114
A[2] =  -2.60662
A[3] =   0.02338
```

Таким образом, искомая эмпирическая зависимость описывается выражением:

$$y = 100.79114 - 2.60662x + 0.02338x^2 \ .$$

При большом числе экспериментальных точек и эмпирических коэффициентов ручное вычисление элементов матрицы системы и даже перечисление их в тексте программы превращается в весьма трудоемкий процесс. Поэтому целесообразно составить программу таким образом, чтобы вычисление элементов матрицы осуществлялось самим компьютером непосредственно по экспериментальной зависимости $y_i(x_i)$. Программа SYS2 решает поставленную выше задачу. В ней экспериментальные значения заданы в виде двух массивов-констант. По ним вычисляются элементы матрицы C_{ij} перед обращением к процедурам MinSqu и LinSys.

```
program SYS2;
      {$N+,E+}
uses Numerics;

const
  M = 7;     { число уравнений = числу эксп. точек }
  N = 3;     { число неизвестных = числу эмпир. коэфф. }
  X: array [1..M] of real =
        ( 7, 12, 17, 22, 27, 32, 37 );
  Y: array [1..M] of real =
        ( 83.7, 72.9, 63.2, 54.7, 47.5, 41.4, 36.3 );
var
  C: array [1..M, 0..N] of extended;
  Matr: array [1..N, 0..N] of extended;
  A: array [1..N] of extended;
  Det: extended;
  i: integer;

BEGIN
      { расчет элементов матрицы Cij }
  for i:= 1 to M do
    begin
      C [i,0]:= Y [i];
      C [i,1]:= 1;
      C [i,2]:= X [i];
      C [i,3]:= Sqr (X [i])
    end;
      { преобразование системы M*N в систему N*N }
```

```
  MinSqu (M, N, C, Matr);
      { решение системы  N*N }
  LinSys (N, Matr, A, Det);
      { вывод эмпирических коэффициентов }
  Writeln ('Эмпирические коэффициенты');
  for i:= 1 to N do
    Writeln  ('A[', i ,'] =', A[i]:9:5)
END.
```

Дальнейшим развитием программы SYS2 является программа SYS3, которая считывает экспериментальные данные из текстового файла 'EXPDATA.DAT' (с ним связана файловая переменная InFile). Разумеется, данные предварительно должны быть записаны в этот файл программой, обслуживающей экспериментальное оборудование. Файл может быть создан и вручную с помощью какого-либо текстового редактора. В нашем случае содержимое файла имеет вид:

```
7      83.7
12     72.9
17     63.2
22     54.7
27     47.5
32     41.4
37     36.3
```

Кроме того, часто желательно знать не только значения эмпирических коэффициентов, но и иметь таблицу значений эмпирической функции (1) для сравнения расчетных значений с экспериментальными значениями y_i. Нередко требуется рассчитать значения эмпирической функции с шагом изменения аргумента меньшим, чем шаг при проведении эксперимента (например, для построения графика эмпирической зависимости).

Программа SYS3 после нахождения эмпирических коэффициентов и вывода их на экран использует эти коэффициенты для вычисления по эмпирической формуле, которая запрограммирована в виде функции с именем Empiric. Вычисления проводятся с помощью цикла while – do, внутри которого независимая переменная Xcalc изменяется от значения X[1] (первая экспериментальная точка) до значения X[M] (последняя экспериментальная точка) с шагом Step. Таблица расчетных значений выводится в текстовый файл с именем CALCDATA.DAT (файловая переменная OutFile).

```
program SYS3;
      {$N+,E+}
uses Numerics;

const
  M = 7;     { число уравнений = числу эксп. точек }
  N = 3;     { число неизвестных = числу эмпир. коэфф. }
  Step = 1; { шаг расчета по эмпирич. формуле }
var
  X, Y: array [1..M] of real;
  InFile, OutFile: text;
  C: array [1..M, 0..N] of extended;
  Matr: array [1..N, 0..N] of extended;
  A: array [1..N] of extended;
  Det: extended;
  i: integer;
  Xcalc: real;

function Empiric (X: real): real;
      { эмпирическая формула }
begin
  Empiric:= A[1] + A[2]*X + A[3]*Sqr(X)
end;  { Empiric }

BEGIN
      { подготовка входного и выходного файлов }
  Assign (Infile, 'EXPDATA.DAT');
  Assign (OutFile, 'CALCDATA.DAT'{'CON'});
  Reset (InFile);
  Rewrite (OutFile);

      { чтение данных и расчет элементов матрицы Cij }
  for i:= 1 to M do
    begin
      Readln (InFile, X[i], Y[i]);
      C [i,0]:= Y [i];
      C [i,1]:= 1;
      C [i,2]:= X [i];
      C [i,3]:= Sqr (X [i])
    end;

      { преобразование системы M*N в систему N*N }
  MinSqu (M, N, C, Matr);
      { решение системы  N*N }
  LinSys (N, Matr, A, Det);

      { вывод эмпирических коэффициентов }
  Writeln ('Эмпирические коэффициенты');
  for i:= 1 to N do
    Writeln  ('A[', i ,'] =', A[i]:9:5);
  Writeln;
```

```
      { расчет по эмпирической формуле }
    Writeln (OutFile, 'X calc':7, 'Y calc':13);
    Writeln (OutFile);
    Xcalc:= X [1];
    while Xcalc < X [M] + Step/2 do
      begin
        Writeln (OutFile, Xcalc:5:0,
                          Empiric (Xcalc):15:3);
        Xcalc:= Xcalc + Step
      end;

 Close (InFile);
 Close (OutFile)
END.
```

Ниже приводятся результаты работы программы SYS3: распечатка файла CALCDATA.DAT. (Эмпирические коэффициенты выводятся программой на экран).

```
 X calc       Y calc

    7         83.690
    8         81.435
    9         79.225
   10         77.063
   ..................
   34         39.194
   35         38.201
   36         37.255
   37         36.355

(часть распечатки опущена).
```

ЗАДАНИЕ

1. В соответствии с предложенным ниже вариантом задания, исходя из эмпирической формулы и таблицы экспериментальных результатов, запишите систему уравнений для вычисления эмпирических коэффициентов методом наименьших квадратов. При необходимости проведите в эмпирической формуле замену переменных.

2. Пользуясь редактором системы Turbo-Pascal или другим текстовым редактором, создайте текстовый файл, содержащий таблицу исходных экспериментальных данных.

3. Составьте программу решения задачи, предусмотрев ввод исходных экспериментальных данных из текстового файла. Программа должна выводить

на экран:

- значения эмпирических коэффициентов,

в текстовый файл с расширением .DAT (в виде таблицы):

- значения независимой переменной,
- рассчитанные по эмпирической формуле значения зависимой переменной.

4. Проведите расчет по составленной программе. По результатам расчета запишите эмпирическую формулу, подставив в нее численные значения эмпирических коэффициентов и, если необходимо, проведя обратную замену переменных.

5. Пользуясь полученным файлом с результатами расчета, постройте графики исследуемых зависимостей с помощью программы EASYPLOT. Экспериментальные результаты изобразите точками, рассчитанные по эмпирической формуле – в виде плавной кривой.

Вариант 1

Вольтамперная характеристика туннельного диода

U, В	*I*, мА
0.025	45
0.05	67
0.075	56
0.1	46
0.125	35
0.15	28
0.175	24
0.2	22
0.225	25
0.25	31
0.275	40
0.3	51
0.325	64
0.35	77
0.375	96

Туннельный диод – полупроводниковый прибор, обладающий удивительным свойством: при увеличении напряжения на его выводах ток через диод уменьшается (в некоторой области напряжений, которая называется *областью отрицательного дифференциального сопротивления*).

При экспериментальном исследовании туннельного диода была получена приведенная здесь таблица зависимости тока, протекающего через диод, от напряжения на нем (так называемая *вольтамперная характеристика*).

Пользуясь методом наименьших квадратов, подберите многочлен пятой степени вида

$$I = a_1 U + a_2 U^2 + a_3 U^3 + a_4 U^4 + a_5 U^5 \ ,$$

описывающий вольтамперную характеристику туннельного диода.

Рассчитайте по найденной формуле ток через диод при изменении напряжения на нем от 0 до 0.375 В с шагом 0.005 В.

Вариант 2

Удельная проводимость полупроводника

Удельной проводимостью вещества называется величина, обратная его удельному сопротивлению. В отличие от металлов, удельная проводимость чистых полупроводников увеличивается с ростом температуры. Температурная зависимость удельной проводимости полупроводника описывается выражением

$$G(T) = G_0 \cdot \mathrm{e}^{-E_g/2kT} \ ,$$

где E_g – энергия, необходимая для того, чтобы электрон в полупроводнике оторвался от атома и получил возможность перемещаться по кристаллу (так называемая *ширина запрещенной зоны полупроводника*), $k = 8.62 \cdot 10^{-5}$ эВ/К – постоянная Больцмана.

T, К	G, $\text{Ом}^{-1}\cdot\text{м}^{-1}$
260	0.25
265	0.39
270	0.51
275	0.65
280	0.85
285	1.2
290	1.4
295	2.0
300	2.6
305	3.1
310	3.9
315	4.8
320	6.1
325	7.5
330	8.8
335	11
340	13
345	15
350	20
355	22

При экспериментальном исследовании этой зависимости была получена приведенная таблица. Исходя из нее, требуется определить значения G_0 и E_g и методом наименьших квадратов, а также рассчитать температурную зависимость удельной проводимости в диапазоне от 250 К до 360 К с шагом 2 К.

Подсказка

Привести эмпирическое выражение к виду, линейному относительно неизвестных коэффициентов, можно путем его логарифмирования и использования замены

$$a_1 = \ln G_0, \quad a_2 = E_g / 2k \ .$$

Вариант 3

Подвижность электронов в полупроводнике

Под действием внешнего электрического поля носители заряда в проводящем материале движутся в среднем с некоторой скоростью, которая называется *дрейфовой скоростью*. Дрейфовая скорость v пропорциональна напряженности электрического поля E.

$$v = \mu \cdot E$$

Коэффициент пропорциональности μ называется *подвижностью* носителей заряда. Подвижность представляет собой скорость дрейфа носителей в поле единичной напряженности.

Теория показывает, что в полупроводниках, содержащих примеси, подвижность носителей меняется с температурой T по закону:

$$\mu = \frac{1}{a_1 T^{3/2} + a_2 T^{-3/2}} \ .$$

T, K	μ, м^2/В·с
80	0.032
100	0.039
120	0.047
130	0.049
140	0.051
160	0.052
180	0.051
200	0.050
205	0.049
220	0.047
240	0.045
250	0.042
295	0.040
300	0.039
310	0.037
340	0.032
390	0.024
430	0.022
500	0.020
610	0.016

При экспериментальном исследовании температурной зависимости подвижности электронов в кремнии с примесью мышьяка была получена приведенная здесь таблица. Пользуясь этой таблицей, определите коэффициенты a_1 и a_2 методом наименьших квадратов, а

также рассчитайте подвижность электронов в диапазоне температур от 50 K до 650 K с шагом 10 K.

Вариант 4

Вольтамперная характеристика варистора

Существуют электронные приборы, сила тока через которые не связана с напряжением на выводах прибора прямо пропорциональной зависимостью. Такие приборы называют *нелинейными*. Один из них – *варистор* (от английских слов *vari*able resi*stor* – изменяемый резистор).

U, В	*I*, мА
11.0	0.1
12.2	0.15
13.5	0.2
17.2	0.3
19.0	0.5
21.5	0.8
22.6	1.0
24.2	1.3
25.5	1.5
27.0	1.8
27.5	2.0
28.0	2.3
29.0	2.5
29.5	2.8
30.0	3.0
30.5	3.4
31.5	3.7
32.0	4.0
32.5	4.2
33.0	4.5

Зависимость силы тока через варистор I от напряжения на его выводах U (вольтамперная характеристика) с достаточной точностью может быть описана выражением

$$I = G_0 \cdot \mathrm{e}^{b\sqrt{U}} \cdot U \ ,$$

где G_0 – проводимость варистора при малых напряжениях, b - параметр нелинейности.

Исходя из приведенной таблицы экспериментальных значений тока через варистор при различных напряжениях, определите значения G_0 и b методом наименьших квадратов. Рассчитайте вольтамперную характеристику варистора в области изменения напряжения от 10 В до 35 В с шагом 0.5 В.

Подсказка

Привести уравнение вольтамперной характеристики к виду, линейному относительно эмпирических коэффициентов, можно путем использования замены

$$a_1 = \ln G_0 \quad , \quad a_2 = b \ .$$

Вариант 5

Градуировочная кривая оптического спектрографа

Для определения длины волны той или иной линии в спектре измеряют ее положение на фотопластинке, на которой сфотографирован спектр. Затем пользуются так называемой *градуировочной кривой спектрографа.* Она связывает координату x спектральной линии с ее длиной волны λ. Градуировочная кривая может быть задана в виде графика, таблицы или в виде формулы.

При градуировке некоторого призменного спектрографа была получена приведенная здесь таблица зависимости положения спектральной линии на фотопластинке от длины волны.

λ, мкм	x, см
0.430	10
0.440	11
0.448	12
0.454	13
0.460	14
0.468	15
0.478	16
0.487	17
0.498	18
0.508	19
0.520	20
0.533	21
0.550	22
0.566	23
0.584	24
0.608	25
0.628	26
0.656	27
0.688	28
0.720	29

Градуировочная кривая призменного спектрографа может быть описана приближенной формулой

$$x = a_1 + \frac{a_2}{\lambda^2} + \frac{a_3}{\lambda^4} + \frac{a_4}{\lambda^6} \ .$$

Методом наименьших квадратов рассчитайте коэффициенты $a_1 \ldots a_4$, пользуясь приведенными в таблице экспериментальными данными. Рассчитайте градуировочную кривую в области длин волн от 0.4 мкм до 0.75 мкм с шагом 0.005 мкм.

Вариант 6

Поглощение света веществом

Исследуется прохождение света через прозрачные плоскопараллельные пластины одного и того же материала различной толщины. В изображенной ниже таблице приведены полученные экспериментально значения коэффициента пропускания света (отношение интенсивности прошедшего света к интенсивности падающего) при различной толщине пластины.

d, мм	τ
0.1	0.7
0.3	0.4
0.5	0.25
0.7	0.15
1.0	0.07
1.3	0.03
1.5	0.02
1.8	0.01
2.0	0.006

С учетом отражения света от поверхностей пластинки зависимость коэффициента пропускания света τ от толщины пластинки d описывается формулой

$$\tau = (1-r)^2 \cdot \mathrm{e}^{-bd} \ ,$$

где r - коэффициент отражения света на границе раздела пластина – воздух, b - *коэффициент поглощения света*.

Исходя из экспериментальных результатов, определите значения r и b для исследуемого материала.

Подсказка

Привести формулу для τ к виду, линейному относительно эмпирических коэффициентов, можно путем логарифмирования и использования замены

$$a_1 = \ln(1-r)^2, \quad a_2 = -b \ .$$

Вариант 7

Температурная зависимость мощности лампы накаливания

Электрическая мощность P, подводимая к нити лампы накаливания, расходуется на нагревание подвеса нити и газа, заполняющего лампу, а также на испускание нитью инфракрасных лучей и видимого света. Пользуясь законами теплопроводности и излучения, можно записать следующее приближенное соотношение

$$P(T) = A \cdot (T - T_0) + \sigma S \cdot T^4 \ ,$$

где

T – температура нити;

T_0 – температура окружающей среды;

A – параметр, характеризующий потери на теплопроводность;

S – эффективная площадь поверхности нити;

$\sigma = 5.67 \cdot 10^{-8}$ Вт·м$^{-2}$·К$^{-4}$ – постоянная Стефана-Больцмана.

Пользуясь методом наименьших квадратов, определите значения параметров A, S, T_0 для лампочки от карманного фонарика, если при экспериментальном исследовании температурной зависимости рассеиваемой лампочкой мощности была получена приведенная здесь таблица. Рассчитайте зависимость $P(T)$ в диапазоне температур от 500 К до 900 К с шагом 50 К.

T, К	P, Вт
560	0.008
580	0.010
620	0.013
650	0.015
730	0.021
800	0.023
880	0.025
960	0.035
1080	0.050
1190	0.063
1290	0.078
1370	0.095
1450	0.113
1530	0.131
1600	0.151
1650	0.173
1710	0.194
1760	0.218
1820	0.241
1870	0.267

*Вариант 8**

Дисперсия света в веществе

Зависимость показателя преломления света вещества от длины световой волны (дисперсия света) может быть объяснена взаимодействием электрического поля волны с упруго связанными электронами, которые входят в состав атомов вещества. Анализ этого процесса в простейшем случае дает следующее выражение для показателя преломления n:

$$n = \sqrt{1 + \frac{Ne^2}{\varepsilon_0 m\left(\omega_0^2 - \omega^2\right)}} \ ,$$

где

N – число электронов в единице объема вещества,

$e = 1.6 \cdot 10^{-19}$ Кл – заряд электрона,

$m = 9.1 \cdot 10^{-31}$ кг – масса электрона,

$\varepsilon_0 = 8.85 \cdot 10^{-12}$ Ф/м – электрическая постоянная,

ω_0 – частота свободных колебаний электрона в атоме,

$\omega = 2\pi c/\lambda$ – частота колебаний поля в световой волне,

$c = 3 \cdot 10^8$ м/с – скорость света в вакууме,

λ – длина световой волны в вакууме.

При исследовании дисперсии света в одном из сортов стекла была получена изображенная здесь таблица зависимости показателя преломления от длины волны. Предполагая, что приведенная выше формула для показателя преломления справедлива, определите методом наименьших квадратов число электронов в единице объема вещества и частоту из свободных колебаний (резонансную частоту). Рассчитайте зависимость $n(\lambda)$ в диапазоне длин волн от 0.22 мкм до 0.5 мкм с шагом 0.01 мкм.

λ, мкм	n
0.22	1.50
0.25	1.31
0.30	1.23
0.35	1.20
0.40	1.17
0.45	1.16
0.50	1.15

Подсказка

Для приведения эмпирической зависимости к виду, линейному относительно эмпирических коэффициентов введите вспомогательную переменную

$$y = 1/(n^2 - 1) \quad .$$

* * *

Приложение

Текст модуля Numerics

```
unit Numerics;

interface
      {$N+,E+,F+}
const
  ArraySize = (2*MaxInt) div SizeOf (extended);

type
  ArrayType = array [1..ArraySize] of extended;

procedure LinSys      { решение системы N линейных
                        алгебраических уравнений с N
                        неизвестными методом Гаусса
                        с выбором ведущего элемента }

   (N: integer;        { число уравнений }
    var Matr;          { расширенная матрица системы }
    var Sol;           { решение системы }
    var Det: extended);{ определитель системы }

procedure MinSqu     { преобразование системы
                       M алгебраических уравнений
                       с N неизвестными
                       в систему N уравнений
                       с N неизвестными
                       методом наименьших квадратов }

     (M: integer;    { число уравнений }
      N: integer;    { число неизвестных }
      var Matr1;     { расширенная матрица системы M*N }
      var Matr2);    { расширенная матрица системы N*N }

      { ========================================== }

implementation

const
  Eps: extended = 0; { начальное значение
                       машинного эпсилон }

      { ------------------------------------------ }

procedure LinSys (N: integer;
                  var Matr; var Sol; var Det: extended);

var
  A: ArrayType absolute Matr;
  X: ArrayType absolute Sol;
  k, l, m: integer;
  C: extended;
```

```
function Ind (i, j: integer): integer;   { внутренняя }
      { вычисляет положение элемента в массиве-строке }
      {      по значениям индексов i,j матрицы A      }
begin
  Ind:= Succ(N)*Pred(i) + Succ(j)
end;  { Ind }

begin
      { вычисление машинного эпсилон при первом вызове }
  if Eps = 0 then
    begin
      Eps:= 1;
      while 1+Eps/2 <> 1 do
        Eps:= Eps/2;
      Eps:= 20*Eps
    end;
      { инициализация решения и определителя }
  for k:= 1 to N do
    X [k]:= 0;
  Det:= 1;
      { прямой ход }
    for k:= 1 to N-1 do
    begin
      C:= Abs (A [Ind (k,k)]);
      l:= k;
      for m:= k+1 to N do
        if Abs (A [Ind (m,k)]) > C then
          begin
            C:= Abs (A [Ind (m,k)]);
            l:= m
          end;

      if C < Eps then
        begin
          Writeln ('НУЛЕВОЙ ОПРЕДЕЛИТЕЛЬ СИСТЕМЫ,');
          Writeln ('ЕДИНСТВЕННОГО РЕШЕНИЯ НЕТ');
          Halt
        end;

      if l <> k then   { перестановка строк }
        begin
          for m:= 0 to N do
            begin
              C:= A [Ind (k,m)];
              A [Ind (k,m)]:= A [Ind (l,m)];
              A [Ind (l,m)]:= C
            end;
          Det:= -Det
        end;

      for m:= k+1 to N do
        begin
          C:= A [Ind (m,k)] / A [Ind (k,k)];
          A [Ind (m,k)]:= 0;
          for l:= k+1 to N do
            A [Ind (m,l)]:= A [Ind (m,l)] - C*A [Ind (k,l)];
```

```
          A [Ind (m,0)]:= A [Ind (m,0)] - C*A [Ind (k,0)]
        end
    end;
      { обратный ход }
  for k:= N downto 1 do
    begin
      C:= 0;
      for l:= k+1 to N do
        C:= C + A [Ind (k,l)] * X [l];
      X [k]:= (A [Ind (k,0)] - C) / A [Ind (k,k)]
    end;
      { вычисление определителя }
  for k:= 1 to N do
    Det:= Det * A [Ind (k,k)]

end;  { LinSys }

      { ---------------------------------------- }

procedure MinSqu (M: integer; N: integer;
                                var Matr1; var Matr2);

var
  C: ArrayType absolute Matr1;
  F: ArrayType absolute Matr2;
  i, j, k: integer;

function Ind (i, j: integer): integer;   { внутренняя }
      { вычисляет положение элемента в массиве-строке }
      {     по значениям индексов i,j матриц A и C    }
begin
  Ind:= Succ(N)*Pred(i) + Succ(j)
end;  { Ind }


begin
  for i:= 1 to N do
    for k:= 0 to i do
      begin
        F [Ind (i,k)]:= 0;
        for j:= 1 to M do
          F [Ind (i,k)]:= F [Ind (i,k)]
                        + C [Ind (j,i)] * C [Ind (j,k)];
        if k > 0 then
          F [Ind (k,i)]:= F [Ind (i,k)]
      end
end;  { MinSqu }

      { ---------------------------------------- }
END.
```

* * *

ОГЛАВЛЕНИЕ

ДЛЯ ЗАМЕТОК

ШКОЛЬНИКУ О КОМПЬЮТЕРНЫХ МЕТОДАХ В ФИЗИКЕ

ПОСОБИЕ-ПРАКТИКУМ ДЛЯ УЧАЩИХСЯ
ВЫПУСКНЫХ КЛАССОВ ШКОЛ И ЛИЦЕЕВ
С УГЛУБЛЕННЫМ ИЗУЧЕНИЕМ
ФИЗИКИ И ИНФОРМАТИКИ

Автор: *Павел Андреевич ВИКТОР*,
преподаватель Ришельевского лицея,
кандидат физико-математических наук,
доцент Одесского национального университета
им. И.И.Мечникова.

Рецензенты:
Анатолий Юрьевич АНИСИМОВ,
зам. директора Одесского областного института
усовершенствования учителей,
Евгений Николаевич КОНДРАТЬЕВ,
кандидат физико-математических наук, доцент.

Книга издана на средства
гранта Президента Украины
для одаренной молодежи.