

*Примеры учат лучше, чем теория.*

Исаак Ньютон

**ЗАДАЧИ ПО  
ПРОГРАММИРОВАНИЮ**

РУБ 39-00

39-00

АМЕЛИНА Н.И., ДЕМЯНЕНКО Я.М., ЛЕБЕДИНСКАЯ Е.Н.,  
ПАСЕЧНЫЙ Л.Г., СПИВАК И.Г., УСОВ А.Б.

**ЗАДАЧИ ПО ПРОГРАММИРОВАНИЮ**

Под редакцией  
доктора физико-математических наук,  
профессора Г.А. Угольницкого

Москва  
«Вузовская книга»  
2000

Амелина Н.И., Демяненко Я.М., Лебединская Е.Н.,  
Пасечный Л.Г., Спивак И.Г., Усов А.Б.  
Задачи по программированию. – М.: Вузовская книга, 2000. – 104 с.  
ISBN -5-89522-109-2

Сборник содержит набор задач, предназначенных для отработки основных приемов программирования. Большую часть книги составляет раздел, содержащий задачи, не ориентированные на какой-либо конкретный язык; соответствующие программы могут быть написаны на том языке, который изучает читатель. Меньшая часть посвящена задачам по языкам Паскаль и Си.

Формулировка большинства задач универсальна в том смысле, что для написания программ могут использоваться разнообразные языки программирования. Поэтому в книге не предлагаются решения большинства задач. Указания к некоторым задачам повышенной трудности и обсуждение некоторых нетрадиционных вопросов включены в текст задач. В начале большинства подразделов приводятся примеры решения типовых задач.

Предлагаемые задачи дают материал для самостоятельных занятий и для практики работы на компьютерах. Диапазон сложности задач довольно широк.

Для начинающих программистов, студентов младших курсов, специализирующихся в области прикладной математики, и для преподавателей, ведущих практические занятия по программированию.

ISBN 5-89522-109-2

© Амелина Н.И., Демяненко Я.М.,  
Лебединская Е.Н., Пасечный Л.Г.,  
Спивак И.Г., Усов А.Б., 2000  
© Оформление «Вузовская книга», 2000

## ГЛАВА I

### ОСНОВНЫЕ ПРИЕМЫ ПРОГРАММИРОВАНИЯ

---

#### 1.1. ПРОСТЕЙШИЕ ПРОГРАММЫ. ОПЕРАТОР ПРИСВАИВАНИЯ. ОПЕРАТОРЫ ВВОДА-ВЫВОДА. ПРОСТЕЙШАЯ ЦЕЛОЧИСЛЕННАЯ АРИФМЕТИКА

*Написать программу, которая вычисляет площадь треугольника по формуле Герона.*

```
Program GERON;
Var a,b,c:real; {стороны треугольника}
      p:real; {полупериметр треугольника}
      s:real; {площадь треугольника}
begin
  writeln('Введите длины треугольника');
  write('a='); read(a);
  write('b='); read(b);
  write('c='); read(c);
  p:=(a+b+c)/2;
  s:=SQRT(p*(p-a)*(p-b)*(p-c));
  writeln('Площадь треугольника равна', s:10:3)
end.
```

1.1.1. Написать программы для решения следующих задач:

- для двух заданных вещественных чисел вычислить и вывести на экран коэффициенты приведенного квадратного уравнения, корнями которого являются эти числа;
- вычислить периметр и площадь правильного шестиугольника, вписанного в окружность заданного радиуса;
- вычислить периметр и площадь прямоугольного треугольника по длинам двух катетов;
- вычислить среднее арифметическое и среднее геометрическое двух положительных чисел;
- по длинам двух сторон треугольника и углу (в градусах) между ними вычислить длину третьей стороны и площадь этого треугольника;

е) по заданным длинам трех сторон треугольника вычислить длины его высот, медиан и биссектрис;

ж) найти координаты точки, делящей в отношении  $n1:n2$  отрезок, заданный координатами своих концов;

з) вычислить длины сторон треугольника по заданным координатам его вершин;

и) вычислить длины медиан треугольника по заданным координатам его вершин.

1.1.2. Для заданного целого четырехзначного числа написать программы решения следующих задач:

+ а) вычислить произведение цифр числа;

- б) вычислить разность между суммой крайних и средних цифр числа;

- в) определить число, полученное выписыванием в обратном порядке цифр данного числа;

- г) вычислить сумму квадратов цифр числа.

## 1.2. АРИФМЕТИКА ВЕЩЕСТВЕННЫХ ЧИСЕЛ. ВЫЧИСЛЕНИЯ ПО ФОРМУЛАМ

Пример вычисления  $y$  по формуле:

$$y = \frac{\sqrt{x} \sin x}{x + e^x}$$

$$y = (\text{SQRT}(x) \sin(x)) / (x + \exp(x))$$

1.2.1. Написать программы вычисления  $y$  по формулам:

а)  $y = \sqrt{x-1} + \frac{1}{x-3};$

(б)  $y = \frac{1}{k\sqrt{2\pi}} e^{-\frac{(x-a)^2}{2k^2}};$

в)  $y = \frac{1}{2b} e^{-\frac{|x-a|}{b}};$

г)  $y = \sqrt{\frac{\pi}{8}} \sqrt{\frac{\sqrt{a^2+b^2}-b}{a^2+b^2}},$

д)  $y = \frac{\sin x}{2\cos^2 x} - \cos x - \frac{3}{2} \operatorname{tg} x;$

е)  $y = \cos x (\ln |2 - e^{-|a+x|}|);$

(ж)  $y = \frac{e^{\sin^2 x} + \ln |\operatorname{arctg} x|}{\sin x},$

з)  $y = ae^{-ax} \sin x;$

и)  $y = \frac{\sin^3 |ax_1^3 + bx_2^2 - ab|}{\sqrt{|ax_1^3 + bx_2^2 - ab|}},$

где  $x_1$  — больший, а  $x_2$  — меньший корень уравнения.

## 1.3. ЛОГИЧЕСКИЙ ТИП. РАЗВЕТВЛЕНИЯ. УСЛОВНЫЙ ОПЕРАТОР. СЕЛЕКТИВНЫЙ ОПЕРАТОР. СОСТАВНОЙ ОПЕРАТОР

Пример программы, реализующей вычисления  $y$  по формуле:

$$y = \begin{cases} \frac{1}{4}, & \text{если } a=b, \\ 2(a^2 + b^2), & \text{если } a>b, \\ \frac{a^2 + b^2}{2}, & \text{если } a<b. \end{cases}$$

```
Program F;
Var a,b,y,x:real;
begin
  write ('Введите значение переменной a=');
  read(a);
  write ('Введите значение переменной b=');
  read(b);
  if a=b then y:=0.25
  else
    begin
      x:=a*a+b*b;
      if a>b then e:=2*x else y:=x/2
    end;
  write ('y=',y)
end.
```

1.3.1. Написать программы вычисления  $y$  по формулам:

a)  $y = \begin{cases} \sin \frac{(a+b)x}{2}, & \text{если } x \neq 0, \\ cb^{ba}, & \text{если } x = 0; \end{cases}$

б)  $y = \begin{cases} x^3 + 3x + 4, & \text{если } x \in [0; 1] \\ (x^3 + 3x + 4)^2, & \text{если } x < 0, \\ -4, & \text{если } x > 0; \end{cases}$

в)  $\begin{cases} z, & \text{если } z \leq 0, \\ 0, & \text{если } 0 \leq z \leq 1, \\ z^2 & \text{если } z > 1; \end{cases}$

где  $z = x^3 + 3x$ ;

г)  $y = \begin{cases} a^2 + b^2, & \text{если } a^2 + b^2 \leq 1, \\ (a+b)\frac{a}{b}, & \text{если } a^2 + b^2 > 1 \text{ и } a \geq b, \\ 0,5, & \text{если } a^2 + b^2 > 1 \text{ и } a < b; \end{cases}$

д)  $y = \begin{cases} \max(a, b), & \text{если } x = 0, \\ \min(a, b), & \text{если } x = 1, \\ |a+b|, & \text{в остальных случаях.} \end{cases}$

1.3.2. Написать программы решения следующих задач:

- а) найти корни квадратного уравнения  $ax^2 + bx + c = 0$ ;  
б) найти решение линейной системы:

$$\begin{cases} a_1 x + b_1 y = c_1, \\ a_2 x + b_2 y = c_2; \end{cases}$$

в) определить максимальное по абсолютной величине из трех заданных чисел  $a, b, c$ ;

г) упорядочить по возрастанию последовательность трех чисел  $a, b, c$ ;

6

д) выяснить, можно ли из отрезков с длинами  $a, b, c$  построить треугольник, и определить тип треугольника;

е) вычислить расстояние от точки плоскости с координатами  $(x, y)$  до границы круга единичного радиуса с центром в начале координат;

ж) выяснить, поместится ли круг площади  $S_1$  в квадрат площади  $S_2$ ;

з) выяснить, пройдет ли кирпич с ребрами  $a, b, c$  в квадратное отверстие со стороной  $d$ ;

и) вычислить расстояние от произвольной точки плоскости  $(x, y)$  до границы квадрата с вершинами  $(0, 0), (0, 1), (1, 1), (1, 0)$ ;

к) переменной  $k$  присвоить номер четверти плоскости, в которой находится точка с заданными координатами  $x$  и  $y$  ( $x, y \neq 0$ );

л) решить уравнение  $ax^4 + bx^2 + c = 0$  при всех возможных значениях  $a, b$  и  $c$ .

1.3.3. Использовать селективный оператор для решения следующих задач:

а) вычислить площадь геометрических фигур

$$S = \begin{cases} ab, & \text{если } n = 1, \\ \frac{ah}{2}, & \text{если } n = 2, \\ \frac{(a+b)h}{2}, & \text{если } n = 3, \\ \pi R^2, & \text{если } n = 4, \\ \frac{\pi R^2 \alpha}{360}, & \text{если } n = 5; \end{cases}$$

б) вычислить площадь

$$S = \begin{cases} Pl, & \text{если } n = 1, \\ \frac{Ph}{2}, & \text{если } n = 2, \\ 2\pi Rh, & \text{если } n = 3, \\ \pi Rl, & \text{если } n = 4, \\ 4\pi R^2, & \text{если } n = 5, \\ \pi R(2h+d), & \text{если } n = 6; \end{cases}$$

в) вычислить  $y$  по формуле

$$y = \begin{cases} a + bx + cx^2, & \text{если } 1 \leq x < 2, \\ (\sin ax)^b, & \text{если } 2 \leq x < 3, \\ \sqrt{|a+bx|} + c, & \text{если } 3 \leq x < 4, \\ a \ln |b+x|, & \text{если } 4 \leq x < 5, \\ e^{ax} + c, & \text{если } 5 \leq x < 6; \end{cases}$$

г) вычислить  $y$  по формуле

$$y = \begin{cases} 1 - \sin x, & \text{если } 5 \leq x < 10, \\ 1 + \cos x, & \text{если } 10 \leq x < 15, \\ \operatorname{tg} x, & \text{если } 15 \leq x < 20, \\ \operatorname{ctg} x, & \text{если } 10 \leq x < 25, \\ 0, & \text{в остальных случаях;} \end{cases}$$

д) определить остаток от деления целой части значения выражения  $\ln|x^2 + ab|$  на 7 и в зависимости от величины выдать сообщение об одном из дней недели.

#### 1.4. ОПЕРАТОРЫ ЦИКЛА. ВЛОЖЕННЫЕ ОПЕРАТОРЫ ЦИКЛА

Примеры вычисления  $k = n!$  с использованием различных операторов цикла:

а) {оператор цикла с предусловием}

$k:=1; i:=2;$

while  $i \leq n$  do

begin  $k:=k*i; i:=i+1$  end;

б) {оператор цикла с постусловием}

$k:=1; i:=2;$

repeat  $k:=k*i; i:=i+1$  until  $i > n$ ;

в) {оператор цикла с параметром}

for  $i := 2$  to  $n$  do  $k:=k*i$

Дано натуральное  $n$  и последовательность чисел  $x_1, x_2, \dots, x_n$ . Вычислить среднее арифметическое членов последовательности.

Program SR;

Var i,n:integer;

```

x,s:real;
begin
write('Введите количество членов последовательности');
read(n);
s:=0;
for i:=1 to n do
begin
    write('Введите очередной член последовательности');
    read(x);
    s:=s+x
end;
s:=s/2;
write('s=',s:10:3);
end.

```

1.4.1. Дано натуральное  $n$  и последовательность целых чисел  $x_1, x_2, \dots, x_n$ . Написать программы решения следующих задач:

а) получить произведение тех членов последовательности, которые нечетны и отрицательны;

б) найти сумму и количество тех членов, которые делятся на 5 и не делятся на 7;

в) вычислить среднее арифметическое всех положительных членов последовательности;

г) выяснить, какое число встречается в последовательности раньше – положительное или отрицательное;

д) определить номер первого четного члена последовательности;

е) определить номер последнего нечетного члена последовательности;

ж) определить количество чисел в наиболее длинной последовательности из подряд идущих нулей;

з) найти наибольшее из четных и количество четных чисел;

и) имеются ли в последовательности два подряд идущих нулевых члена.

1.4.2. Дано натуральное  $n$  и последовательность вещественных чисел  $x_1, x_2, \dots, x_n$ :

а) получить  $a_i = \frac{x_i}{1 + (a_1 + a_2 + \dots + a_i)^2}, i = 1, 2, \dots, n$ ;

б) проверить, упорядочена ли последовательность по неубыванию или по невозрастанию;

в) найти  $k$ , при котором  $x_{k-1} < a < x_k$ ;

г) вычислить количество тех членов последовательности, для которых выполнено  $i-1 < x_i < i$ ;

д) найти длину наименьшего отрезка числовой оси, содержащего числа  $x_1, x_2, \dots, x_n$ ;

е) определить число соседств двух положительных чисел;

ж) найти произведение членов последовательности, расположеннымми между первым и вторым нулевыми членами.

1.4.3. Найти первый член последовательности, для которого выполнено условие  $|a_n - a_{n-1}| < \varepsilon$ , если последовательность образована по закону:

$$a) a = \frac{n}{\sqrt{n^2-1} + \sqrt{n^2+1}};$$

$$б) a_n = \left(1 - \frac{1}{2}\right)\left(1 - \frac{1}{3}\right) \dots \left(1 - \frac{1}{n+1}\right);$$

$$в) a_n = \left(1 - \frac{1}{2!}\right)\left(1 + \frac{1}{3!}\right) \dots \left(1 + \frac{(-1)^n}{(n+1)!}\right);$$

$$г) a_1 = x, a_n = \sqrt{14a_{n-1}^2 - 2x};$$

$$д) a_1 = x, a_n = 2a_{n-1} + \frac{x}{4 + a_{n-1}^2};$$

$$е) a_1 = x, a_n = 3 + \frac{1}{2^n} \cos^2(a_{n-1} - x).$$

1.4.4. Написать программы вычисления значения  $Y$  для заданного натурального  $n$ :

$$а) Y = \sum_{k=1}^n \frac{1}{k^2};$$

$$б) Y = \sum_{k=1}^n k^k;$$

$$в) \textcircled{2} Y = \sum_{k=1}^n \frac{1}{k!};$$

$$г) Y = \sum_{k=1}^n \frac{1}{(2k)^2};$$

$$д) Y = \sum_{k=1}^n \frac{1}{k\sqrt{k}};$$

$$е) Y = \sum_{k=1}^n \frac{(-1)^k}{(2k+1)^k};$$

$$ж) Y = \sum_{k=1}^n \frac{(-1)^k}{k(k+1)};$$

$$з) Y = \sum_{k=1}^n \frac{(-1)^k (k+1)}{k!};$$

$$и) Y = \sum_{k=1}^n \left(2 + \frac{1}{k!}\right);$$

$$к) Y = \sum_{k=2}^n \left(1 - \frac{1}{k!}\right)^2.$$

1.4.5. Написать программы вычисления  $Y$  для заданного натурального  $n$  и вещественного  $x$ :

$$а) Y = \sum_{k=1}^n \frac{x^k}{k!};$$

$$б) Y = \sum_{k=1}^n \frac{1}{k!} + \sqrt{|x|};$$

$$в) Y = \sum_{k=1}^n \frac{x + \cos x}{2^k};$$

$$г) Y = \sum_{k=1}^n \frac{(-1)^k x^{2k}}{k(k+1)(k+2)};$$

$$д) Y = \sum_{k=1}^n \frac{x^{2k} \sin(x^k)}{k^2};$$

$$е) Y = \prod_{k=1}^n \left(1 + \frac{\sin(kx)}{k!}\right);$$

$$ж) Y = \prod_{k=1}^n \left(\frac{k}{k+1} - \cos^k |x|\right);$$

$$з) Y = \prod_{k=1}^n \left(\frac{(1-x)^{k+1} + 1}{((k-1)! + 1)^2}\right).$$

1.4.6. Для заданного натурального числа  $n$  составить программы решения следующих задач:

а) подсчитать  $k$  — количество цифр в десятичной записи числа  $n$ ;

б) выяснить, является ли число  $n$  степенью числа 3;

в) найти сумму цифр числа  $n$ ;

г) определить число, получаемое выписыванием в обратном порядке цифр числа  $n$ ;

д) выяснить, является ли заданное число палиндромом, т.е. таким, десятичная запись которого читается одинаково слева направо и справа налево.

1.4.7. Написать программы вычисления  $Y$  по формулам:

$$а) Y = \sum_{i=1}^n \left(\frac{1}{i}\right)^n;$$

$$б) Y = \sum_{i=1}^n \left(\frac{1}{i}\right)^i;$$

$$в) Y = \sum_{i=1}^n \left(\frac{1}{i}\right)^{n-i+1};$$

$$г) Y = \prod_{i=1}^n \left(1 + \frac{1}{i^n}\right);$$

$$\textcircled{2} д) Y = \prod_{i=1}^n \left(1 + \frac{1}{i^i}\right);$$

$$е) Y = \prod_{i=1}^n \left(1 + \frac{1}{i}\right)^n;$$

$$ж) Y = \prod_{i=1}^n \left(1 + \frac{1}{i}\right)^i;$$

$$з) Y = \prod_{i=1}^n \left(1 + \frac{1}{i}\right)^{n-i+1};$$

и)  $Y = \sum_{i=1}^n \sum_{j=1}^k (a^i + x^j);$

к)  $Y = \prod_{i=0}^n \prod_{j=1}^k \left( j + \frac{x}{i+2} \right);$

л)  $Y = \sum_{i=1}^n \sum_{j=1}^k x^{\frac{i}{j}},$  для  $x = a, a+h, a+2h, \dots, b,$  где  $h = (b-a)/m,$

$b > a,$

м)  $Y = \prod_{i=1}^n \prod_{j=1}^k \frac{i}{1+x^{i+j}},$  для  $x = a, a+h, a+2h, \dots, b.$

## 1.5. ВЫЧИСЛЕНИЯ С ЗАДАННОЙ ТОЧНОСТЬЮ

Вычислить с заданной точностью  $\epsilon$  значения функции  $y = f(x)$ , заданной рядом  $y = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$

```
Program SUM1;
Var x,y,EPS,R:real;
    K:integer;
begin
  write('Введите значение аргумента x=');
  read(x);
  write('Введите точность вычисления EPS=');
  read(EPS);
  y:=0; R:=1; k:=1;
  while ABS(R)>=EPS do
    begin
      y:=y+R;
      R:=R*x/k;
      k:=k+1;
    end;
  write('y=',y:10:3)
end.
```

1.5.1. Написать программы вычисления с заданной точностью следующих сумм:

а)  $y = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \dots$

б)  $y = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots$

в)  $y = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \dots$

г)  $y = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \dots$

д)  $y = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \dots$

е)  $y = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \dots,$  для  $-1 < x \leq 1$

ж)  $y = x - \frac{x^3}{3} + \frac{x^5}{5} - \dots,$  для  $|x| \leq 1$

з)  $y = x + \frac{x^3}{3} + \frac{x^5}{5} + \dots,$  для  $|x| < 1$

и)  $y = x - \frac{1}{2} \cdot \frac{x^3}{3} + \frac{1 \cdot 3}{2 \cdot 4} \cdot \frac{x^5}{5} - \frac{1 \cdot 3 \cdot 5}{2 \cdot 4 \cdot 6} \cdot \frac{x^7}{7} + \dots,$  для  $|x| \leq 1$

к)  $y = 1 + m x + \frac{m(m-1)}{2!} x^2 + \frac{m(m-1)(m-2)}{3!} x^3 + \dots,$

для любого действительного  $m$  и  $|x| < 1$

## 1.6. СИМВОЛЬНЫЙ ТИП. ОБРАБОТКА ПОСЛЕДОВАТЕЛЬНОСТЕЙ СИМВОЛОВ

1.6.1. Для данного натурального  $n$  и последовательности символов  $s_1, s_2, \dots, s_n$  составить следующие программы:

- а) подсчитать общее число вхождений символов  $+, -, *, /;$
- б) выяснить, встречаются ли в данной последовательности группы из трех стоящих рядом точек;
- в) получить первое  $i$ , для которого  $s_i$  и  $s_{i+1}$  совпадают с буквой  $a;$
- г) найти такое  $i$ , что  $s_i$  — первая по порядку запятая;
- д) выдать все пары совпадающих одинаковых символов;
- е) подсчитать наибольшее число букв  $a$ , идущих подряд;
- ж) определить общее число латинских букв, входящих в последовательность.

1.6.2. Для заданной последовательности символов, за которой следует точка (в сам текст точки не входит), составить программы:

- переменной  $f$  присвоить значение *true*, если буква  $a$  встречается чаще, чем буква  $b$ , и присвоить значение *false* в противном случае;
- подсчитать, сколько раз встречается слово *key*;
- проверить, правильно ли в данной последовательности расположены круглые скобки;
- определить, является ли данная последовательность символов правильной записью целого числа (возможно, со знаком).

1.6.3. Последовательность символов содержит слова из латинских букв, соседние слова отделены друг от друга запятой, за последним символом — точка. Написать программы решения задач:

- определить общее число символов;
- определить количество слов, которые начинаются с буквы  $a$ ;
- определить количество слов, которые оканчиваются буквой  $i$ ;
- определить количество слов, которые начинаются и оканчиваются одной и той же буквой.

## 9 1.7. ПРОЦЕДУРЫ. ФУНКЦИИ

Описать процедуру, определяющую значение суммы двух комплексных чисел, представленных в алгебраической форме.

```
Procedure CADD(Re1, Im1, Im2: real; var Re, Im: real);  
begin  
    Re:=Re1+Re2;  
    Im:=Im1+Im2  
end;
```

1.7.1. Описать процедуры, реализующие арифметические операции над комплексными числами, представленными в алгебраической форме:

- определить разность двух комплексных чисел;
- определить произведение двух комплексных чисел;
- определить частное от деления двух комплексных чисел, если оно существует.

1.7.2. Описать процедуры, которые переводят комплексное число из арифметической формы представления в тригонометрическую и наоборот.

1.7.3. Описать процедуры, выполняющие следующие действия

над комплексными числами, заданными в тригонометрической форме:

- умножение двух комплексных чисел;
- деление двух комплексных чисел;
- возвведение комплексного числа в степень с натуральным показателем  $n$ .

1.7.4. Написать программы решения следующих задач:

- для заданного комплексного числа, представленного в алгебраической форме, и числа  $n$  определить все значения  $\sqrt[n]{c}$ ;
- определить корни уравнения  $ax^2 + bx + c = 0$  с комплексными коэффициентами  $a, b, c$ ;
- определить корни приведенного кубического уравнения  $x^3 + ax^2 + bx + c = 0$  с действительными коэффициентами.

Описать функцию, вычисляющую сумму квадратов первых  $n$  натуральных чисел.

```
Function SN2(n:integer):integer;  
Var s,i:integer;  
begin  
    s:=0  
    for i:=1 to n do  
        s:=s+i*i;  
    SN2:=s  
end.
```

1.7.5. Описать функции для решения следующих задач:

- для заданного значения  $x$  и точности  $\varepsilon$  вычислить сумму ряда  $x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots$ ;
- для заданного значения  $x$  и точности  $\varepsilon$  вычислить произведение  $x \times \prod_{k=1}^{\infty} \left(1 - \left(\frac{x}{k \times \pi}\right)^2\right)$ ;
- для заданного вещественного  $x$  и натурального  $n$  вычислить  $x^n$ ;
- для заданных натуральных  $n$  и  $m$  вычислить биноминальный коэффициент, используя формулу  $C_n^m = \frac{n!}{m!(n-m)!}$ ;
- для заданных натуральных  $n$  и  $m$  вычислить биноминальный коэффициент, используя определение

$$C_n^m = \begin{cases} 1, & \text{если } m=0 \text{ или } m=n, \\ C_{n-1}^m + C_{n-1}^{m-1}, & \text{если } n>m>0; \end{cases}$$

ж) для заданных координат двух точек плоскости вычислить расстояние между ними;

з) вычислить расстояние от точки  $(x, y)$  до прямой  $Ax + By + C = 0$ ;

и) вычислить угол в радианах между прямыми  $A_1 x + B_1 y + C_1 = 0$  и  $A_2 x + B_2 y + C_2 = 0$ ;

к) определить координаты точки пересечения двух прямых  $A_1 x + B_1 y + C_1 = 0$  и  $A_2 x + B_2 y + C_2 = 0$  и вернуть значение *true*, если точка существует, или значение *false* в противном случае.

1.7.6. Составить программы решения следующих задач:

а) вычислить  $Y = \frac{\operatorname{th}(a+b)}{\operatorname{th}(a+c)} - \frac{\operatorname{th}(c+b)}{\operatorname{th}(a-c)}$ , используя для вычисления

приближенного значения гиперболического тангенса одну из формул:

$$\operatorname{th} x = \frac{e^x - e^{-x}}{e^x + e^{-x}}; \quad \operatorname{th}(x) = \frac{\sum_{k=0}^{\infty} \frac{x^{2k+1}}{(2k+1)!}}{\sum_{k=0}^{\infty} \frac{x^{2k}}{(2k)!}};$$

б) для заданного вещественного положительного  $a$  вычислить величину  $\frac{\sqrt[3]{a} - \sqrt[6]{a^2+1}}{1 + \sqrt[3]{a+1}}$ ,

в) используя для вычисления  $y = \sqrt[k]{x}$  с точностью  $\epsilon$  итерационную формулу:

$$y_0 = 1, y_{n+1} = y_n + \left( \frac{x}{y_n^{k-1}} - y_n \right) / k;$$

для заданных координат вершин двух треугольников определить тот, который имеет наибольшую площадь.

## 1.8. ПАРАМЕТРЫ-ПРОЦЕДУРЫ И ПАРАМЕТРЫ-ФУНКЦИИ

Написать программу вычисления интеграла  $w = \int_a^b \frac{\sin x}{1+x^2} dx$  по формуле правых прямоугольников.

16

```
Program P;
type FUNC=function(x:real):real;
Var a,b,w,eps:real;
function F(x:real):real;far;
begin
  F:=sin(x)/(1+x*x)
end;
function INT(a,b,eps:real;F:FUNC):real;
Var i,n:real;
  x,h,S1,S2:real;
begin
  {первоначальное вычисление площади}
  n:=1; h:=(b-a)/n;
  S2:=f(a+h/2)*h;
  {цикл по уточнению площади}
  repeat
    S1:=S2; n:=2*n; h:=h/2;
    S2:=0;
    X:=a+h/2;
    for i:=1 to n do
      begin
        S2:=S2+F(x);
        X:=X+h
      end;
    S2:=S2*h
  until ABS(S2-S1)<=eps;
  INT:=S2
end;
begin
  writeln('a,b=?'); readln(a,d);
  writeln('eps=?'); readln(eps);
  w:=INT(a,b,eps,F);
  writeln('w=',w:10:4)
end.
```

1.8.1. Написать программы вычисления с заданной точностью интеграла  $w = \int_a^b f(x) dx$ ,

а) по формуле левых прямоугольников  $S \approx h \sum_{i=0}^{n-1} f(a+ih)$ ,

$$h = \frac{b-a}{n};$$

17

6) по формуле трапеций  $S \approx h \left( \frac{f(a) + f(b)}{2} + \sum_{i=1}^{n-1} f(a + ih) \right)$ .

1.8.2. Написать программы вычисления с заданной точностью  $\varepsilon$  корня уравнения  $F(x) = 0$  на заданном отрезке  $[a, b]$ , используя один из описанных ниже методов.

а) Метод простых итераций основан на представлении уравнения  $F(x) = 0$  в виде  $x = f(x)$  и многократном применении итерационной формулы  $x_{n+1} = f(x_n)$  до тех пор, пока соблюдается условие  $|x_{n+1} - x_n| \geq \varepsilon$ . Итерационный процесс сходится, если соблюдается условие  $f'(x) < 1$  при  $a < x < b$ .

б) Метод касательных (Ньютона) основан на замене  $F(x)$  в точке начального приближения  $x = x_0$  касательной, пересечение которой с осью  $x$  дает первое приближение  $x_1$  и т.д. Таким образом, итерационный процесс схождения к корню реализуется формулой  $x_{n+1} = x_n - \frac{F(x_n)}{F'(x_n)}$  до тех пор, пока соблюдается условие  $|x_{n+1} - x_n| \geq \varepsilon$ .

Метод обеспечивает быструю (квадратичную) сходимость, если  $F(x_0) F''(x_0) > 0$ . В качестве  $x_0$  выбирают тот конец отрезка  $[a, b]$ , на котором знаки  $F(x_0)$  и  $F''(x_0)$  совпадают.

в) Метод хорд. При этом методе каждое значение  $x_{n+1}$  находится как точка пересечения оси абсцисс с хордой, проведенной через точки  $F(a)$  и  $F(b)$ , причем одна из этих точек фиксируется – та, для которой знаки  $F(x)$  и  $F''(x)$  одинаковы.

Если неподвижен конец хорды  $x = a$ , то  $x_{n+1} = x_n - \frac{F(x_n)}{F(x_n) - F(a)}(x_n - a)$ . Если неподвижен конец хорды  $x = b$ , то  $x_{n+1} = x_n - \frac{F(x_n)}{F(b) - F(x_n)}(b - x_n)$ .

Если  $|x_{n+1} - x_n| \geq \varepsilon$ , то в первом случае считаем  $b = x_{n+1}$ , во втором  $a = x_{n+1}$  и повторяем вычисления.

## 2.1 МАССИВ КАК СТРУКТУРА ДАННЫХ. ВЕКТОРЫ

Дано 100 целых чисел. Напечатать сначала все отрицательные числа, а затем – все остальные.

```
Program mas100;
const n=100;
type IArray=array[1..n] of integer;
var A:IArray;
i:byte;
begin
writeln('Введите элементы массива.');
for i:=1 to n do readin(A[i]);
writeln('Печать по заданному правилу:');
for i:=1 to n do
  if A[i]<0 then writeln(A[i]);
for i:=1 to n do
  if not (A[i]<0) then writeln(A[i]);
end.
```

2.1.1. Дан массив целых чисел. Произвести для него следующие операции:

- выдать все числа, входящие в массив по одному разу;
  - найти число различных элементов массива;
  - выяснить, имеется ли в массиве хотя бы одна пара совпадающих чисел;
  - для каждого из чисел, входящего в массив, указать, сколько раз оно входит в массив;
  - определить количество элементов в наиболее длинной группе из подряд идущих нулей;
  - определить, сколько раз в массиве меняется знак;
  - определить количество инверсий в массиве (т.е. таких пар элементов, в которых большее число находится слева от меньшего).
- 2.1.2. Дан  $A$  — массив из вещественных чисел. Для него нужно:
- вычислить

$$y = \sum_{i=1}^n (-1)^i A_i$$

без использования операции возведения в степень;

б) получить массив  $B$ , в котором  $i$ -й элемент является средним арифметическим всех элементов массива  $A$ , кроме  $i$ -го.

2.1.3. Дан массив вещественных чисел. Найти:

а) произведение и среднее арифметическое элементов массива, предшествующих первому нулевому элементу;

б) сумму и произведение элементов массива, расположенных между первым и вторым нулевыми элементами;

в) количество и сумму положительных элементов массива, предшествующих первому отрицательному элементу;

г) среднее арифметическое первых подряд идущих положительных элементов.

2.1.4. Элементы массива (числового или символьного)

а) расположить в обратном порядке;

б) циклически сдвинуть на одну позицию влево (вправо);

в) циклически сдвинуть на  $k$  позиций влево (вправо).

2.1.5. В массиве найти наибольший (наименьший) элемент и поменять его местами с первым (последним) элементом.

2.1.6. Из массива удалить:

а) все элементы, равные заданному значению;

б) первый из нулевых элементов;

в) элементы, принадлежащие заданному интервалу.

2.1.7. Преобразовать массив вещественных чисел по правилу:

а) положительные элементы заменить их квадратами, отрицательные – абсолютными величинами;

б) домножить все отрицательные элементы массива на первый, если он тоже отрицателен;

в) все отрицательные (положительные) элементы перенести в начало массива, а остальные – в конец, сохранив порядок их следования;

г) минимальный элемент заменить целой частью среднего арифметического всех элементов массива (если в массиве несколько элементов с минимальным значением, заменить последний по порядку);

д) все элементы с нечетными номерами, предшествующие первому по порядку максимальному элементу, домножить на значение максимального элемента;

е) если первый элемент неотрицателен, домножить все элементы

на квадрат минимального элемента, в противном случае – на квадрат максимального;

ж) элементы массива циклически сдвинуть на  $k$  позиций влево.

2.1.8. Дан массив вещественных чисел. Определить:

а) отношение суммы элементов массива, расположенных до максимального элемента в массиве, к сумме элементов, расположенных после максимального;

б) произведение элементов массива, расположенных между максимальным и минимальным элементами;

в) отношение суммы элементов массива, расположенных до минимального элемента в массиве, к их произведению;

г) среднее арифметическое элементов массива, расположенных между максимальным и «центральным» элементами массива (предполагается, что в массиве нечетное число элементов).

2.1.9. Вычислить скалярное произведение двух векторов.

2.1.10. Для заданного одномерного массива (вектора) с компонентами вещественного типа написать программы решения задач:

а) определить значение наибольшего элемента и его порядковый номер в массиве;

б) вычислить разность между наибольшим и наименьшим элементом;

в) выяснить, упорядочены ли элементы массива по невозрастанию или неубыванию;

г) осуществить циклический сдвиг элементов вектора на  $k$  позиций вправо (влево);

д) вычислить длину вектора по формуле:  $d = \sqrt{a_1^2 + a_2^2 + \dots + a_n^2}$ , где  $n$  – количество элементов массива;

е) вычислить среднее арифметическое его положительных (отрицательных) элементов;

ж) вычислить произведение его положительных элементов;

з) переставить элементы массива так, чтобы в начале массива располагались все отрицательные, а в конце массива все положительные элементы;

и) расположить элементы массива в обратном порядке;

к) преобразовать массив по правилу  $x'_k = \max x_i$ , при  $1 \leq i \leq k$ ;

л) вычислить сумму тех элементов массива, которые расположены между наибольшим и наименьшим элементом;

м) определить количество инверсий элементов массива, т.е. таких пар элементов, что  $x_i > x_j$ , при  $i < j$ .

В задачах № 2.1.11–2.1.14 под текстом понимается массив, элементами которого являются символы.

2.1.11. Дан текст из 80 литер. Напечатайте вначале все цифры, входящие в этот текст, а затем все остальные литеры, сохраняя при этом взаимное расположение литер в каждой из этих двух групп.

2.1.12. Дан текст, содержащий от 1 до 70 букв, за которым следует точка. Напечатайте этот текст в обратном порядке.

2.1.13. Дан непустой текст из цифр, за которыми следует точка. Напечатайте этот текст в обратном порядке и укажите наиболее часто встречающуюся цифру.

2.1.14. Дан текст из 80 литер. Определить, симметричен ли он, т.е. читается ли он одинаково справа налево и наоборот.

2.1.15. Даны два вещественных массива с разным числом элементов. Найти:

- а) общее число нулевых элементов массивов;
- б) минимальную из сумм положительных элементов массивов;
- в) сумму минимальных элементов массивов;
- г) отношение максимальных элементов массивов;
- д) модуль разности чисел, являющихся минимальными среди положительных элементов массивов.

2.1.16. Даны вещественные векторы А и В с разным числом элементов. Логической переменной Flag присвоить значение «истина», если:

- а) у массива А больше положительных элементов, чем у массива В;
- б) сумма отрицательных элементов массива А меньше суммы отрицательных элементов массива В;
- в) в массиве А нулевых элементов меньше, чем в массиве В;
- г) произведение ненулевых элементов массива А больше произведения ненулевых элементов массива В;

иначе Flag положить равной «ложь». Значение переменной Flag вывести на печать.

2.1.17. Даны вещественные векторы X и Y. Преобразовать каждый из них по правилу:

- а) если у вектора есть хотя бы один элемент, принадлежащий отрезку  $[0, 1]$ , то все отрицательные элементы заменить нулями. В противном случае – единицами;
- б) заменить на нуль элементы, предшествующие первому по порядку отрицательному элементу (если отрицательных элементов нет, вектор оставить без изменений);

в) всем элементам, следующим за максимальным элементом, присвоить значение максимального.

## 2.2. МАТРИЦЫ

2.2.1. Для каждой из трёх заданных квадратных матриц определить

- а) след матрицы, то есть сумму диагональных элементов;
- б) норму матрицы по формуле

$$N = \max_{1 \leq i \leq m} \sum_{j=1}^m$$

2.2.2. Найти для каждой из трёх заданных прямоугольных матриц

- а) сумму квадратов её элементов;
- б) произведение положительных элементов;
- в) минимальный элемент;
- г) максимальный элемент и номера строки и столбца, на пересечении которых он находится.

2.2.3. Даны три прямоугольные матрицы. Получить из каждой новую матрицу путём

- а) деления всех элементов на квадрат последнего элемента последней строки;
- б) замены всех элементов, по абсолютной величине больших X, на X;
- в) замены нулями всех отрицательных элементов.

2.2.4. Каждую из трёх прямоугольных матриц преобразовать по правилу:

- а) если последний элемент строки отличен от нуля, разделить на него все элементы этой строки;
- б) если первый элемент столбца отрицателен, возвести в квадрат все элементы этого столбца;
- в) разделить все элементы матрицы на элемент, наибольший по абсолютной величине;
- г) домножить все элементы матрицы на минимальный элемент этой матрицы;
- д) все элементы каждой строки матрицы разделить на минимальный элемент этой строки;

е) все элементы каждого столбца матрицы умножить на максимальный элемент этого столбца.

2.2.5. Даны три прямоугольные матрицы. Получить для каждой из них вектор, каждая компонента которого:

- сумма всех элементов соответствующей строки матрицы;
- произведение всех элементов соответствующего столбца матрицы;
- сумма отрицательных элементов соответствующего столбца;
- произведение отрицательных элементов соответствующей строки;
- количество нулевых элементов соответствующей строки матрицы;
- число отрицательных элементов соответствующего столбца матрицы;
- минимальный элемент соответствующего столбца матрицы;
- максимальный элемент соответствующей строки матрицы;
- наибольший по абсолютной величине элемент соответствующей строки матрицы.

2.2.6. Для заданного двумерного массива (матрицы) с компонентами вещественного типа написать программы решения задач:

- определить минимальный и максимальный элементы;
- поменять местами строки с номерами  $i$  и  $j$ ;
- определить количество различных элементов матрицы, т.е. повторяющиеся элементы считать один раз;
- определить, является ли матрица ортонормированной, т.е. такой, в которой скалярное произведение каждой пары различных строк равно 0, а скалярное произведение каждой строки на себя равно 1;
- определить, является ли заданная квадратичная матрица магическим квадратом, т.е. такой, в которой суммы во всех строках и столбцах одинаковы;
- определить индексы всех седловых точек матрицы, т.е. таких элементов, которые являются наименьшими в своей строке и одновременно наибольшими в своем столбце.

2.2.7. Получить таблицу Пифагора – квадратную матрицу размером  $10 \times 10$ , элементы которой определяются формулой

$$P_{ij} = i \cdot j.$$

2.2.8. По заданной матрице получить новую, каждый элемент ко-

торой получается как среднее арифметическое имеющихся соседей соответствующего элемента исходной матрицы. Соседями элемента  $A_{ij}$  матрицы называют элементы  $A_{kl}$ , для которых

$$i - 1 \leq k \leq i + 1, \quad j - 1 \leq l \leq j + 1, \quad (k, l) \neq (i, j).$$

2.2.9. Подсчитать количество локальных минимумов заданной матрицы.

2.2.10. Найти максимум среди всех локальных минимумов заданной матрицы.

2.2.11. Определить, является ли заданная целая квадратная матрица:

- симметричной относительно главной диагонали;
- ортонормированной, т.е. такой, в которой скалярное произведение каждой пары различных строк равно 0, а скалярное произведение каждой строки на себя равно 1;
- магическим квадратом, т.е. такой, в которой суммы элементов во всех строках и столбцах одинаковы.

2.2.12. Построчно вводится прямоугольная матрица. Преобразуйте матрицу, переставляя её строки и столбцы так, чтобы наибольший элемент оказался в правом нижнем углу.

2.2.13. Найти сумму двух прямоугольных матриц.

2.2.14. Найти произведение вектора на матрицу и произведение матрицы на вектор.

2.2.15. Найти произведение двух прямоугольных матриц.

2.2.16. Даны натуральное  $n$  и элементы квадратной вещественной матрицы четвёртого порядка, которые вводятся по строкам или столбцам. Вычислить  $n$ -ю степень этой матрицы.

2.2.17. Получить матрицу, транспонированную по отношению к заданной прямоугольной матрице.

2.2.18. Проверить, является ли матрица  $B$  обратной для матрицы  $A$ .

2.2.19. Данна квадратная матрица 5-го порядка. Найти матрицу, обратную ей, или установить, что такой матрицы не существует.

2.2.20. Для заданной целой матрицы размером  $5 \times 7$  напечатать индексы всех ее седловых точек. Седловой точкой считать элемент, являющийся наименьшим в своей строке и одновременно наибольшим в своем столбце или наибольшим в строке, но наименьшим в столбце.

## 2.3. ВЫЧИСЛЕНИЯ С ХРАНЕНИЕМ ПОСЛЕДОВАТЕЛЬНОСТИ ЗНАЧЕНИЙ

2.3.1. Компания платит своим продавцам на комиссионной основе. Продавцы получают 200 рублей в неделю плюс 9% от валовой продажи за неделю (зарплата сотрудников округляется до целого значения). Используя массив счётчиков, напишите программу, которая определяет, сколько продавцов получили заработную плату в каждом из следующих диапазонов: 200–299 руб., 300–399 руб., ..., 900–999 руб., 1000 и более.

2.3.2. По заданным вещественным числам  $a_0, a_1, \dots, a_{20}$  и  $t$  вычислить значение многочлена

$$a_{20} x^{20} + a_{19} x^{19} + \dots + a_1 x$$

и его производной в точке  $t$ .

2.3.3. По заданным коэффициентам  $A_{ij}$  и правым частям  $b_i$  решить систему линейных уравнений.

$$\sum_{j=1}^n A_{ij} x_j = b_i.$$

Считать, что определитель матрицы отличен от нуля.\*

2.3.4. Напишите программу, которая вводит значения ежемесячной зарплаты для заданного двенадцатимесячного периода и затем определяет месяц, в котором

- обнаружен наибольший прирост зарплаты;
- обнаружено наибольшее уменьшение зарплаты.

2.3.5. Даны таблица результатов некоторого спортивного турнира, в котором участвовало 10 спортсменов. Элементы таблицы записаны по следующему правилу: если  $i$ -й спортсмен выиграл у  $j$ -го, то в  $i$ -й строке и  $j$ -м столбце ставится 'V', в случае проигрыша – 'P' в соответствующей клетке таблицы, при ничьей – 'N', а на главной диагонали записываются 'X'. За выигрыши даётся одно очко, за ничью – полочка, за поражение – ноль. Выдать список участников в порядке убывания набранных ими очков.

2.3.6. Напишите программу кодирования текстов методом Цезаря, в котором каждую букву алфавита заменяет другая определенная буква, например «а» вместо «б», «б» вместо «ж» и т.д.

\* Рекомендация: вначале приведите систему уравнений к треугольному виду.

2.3.7. Напишите программу, подбирающую ключ к коду Цезаря (смотрите предыдущую задачу): для этого подсчитайте частоты встречаемости букв в закодированном тексте и частоты встречаемости букв в русском языке.

2.3.8. Напишите программу кодирования текстов кодом Виженера. В нем несколько раз производится регулярная замена кода Цезаря:  $k$ -я буква сообщения зашифровывается  $(k \text{ MOD } n)$ -м алфавитом, причём  $n$  – это общее число применяемых алфавитов.

## 2.4. СОРТИРОВКА МАССИВОВ

Методы сортировки можно разбить на три основные группы: сортировка выбором, сортировка обменом, сортировка вставками.

Для сортировки  $n$ -элементного массива методом прямого выбора отыскивается максимальный (минимальный) элемент и меняется местами с последним (первым) элементом. Затем этот процесс повторяется с оставшимися  $n-1$  элементами,  $n-2$  элементами и т.д.

Простейший алгоритм прямого обмена — «метод пузырька» — основан на сравнении пары соседних элементов и их обмене, если они расположены не по порядку, и продолжении этого процесса до тех пор, пока не будут упорядочены все элементы.

При сортировке вставками элементы просматриваются по одному, начиная со второго, каждый очередной  $i$ -й элемент вставляется в подходящее место среди  $(i-1)$ -го ранее упорядоченного элемента.

Процесс нахождения элемента массива, значение которого равно заданному ключевому значению, называется поиском. Если для проведения процедуры поиска не предлагается дополнительной информации, то используется простейший метод линейного поиска, при котором последовательно перебираются все элементы массива до тех пор, пока не будет обнаружен первый ключевой элемент, или пока весь массив не будет просмотрен, а совпадений найти не удастся.

Для упрощения процедуры поиска используется метод линейного поиска с барьером. В конец массива помещается дополнительный  $(n+1)$ -й элемент, значение которого кладётся равным ключевому. Далее действия производятся в соответствии с алгоритмом линейного поиска. Данный метод гарантирует, что ключевой элемент будет найден.

Поиск можно сделать более эффективным, если элементы массива будут упорядочены. В этом случае применяется алгоритм двоичного поиска. Алгоритм определяет местоположение среднего элемента

массива и сравнивает его с ключевым. Если они равны, то ключевой элемент найден и выдаётся его индекс. В противном случае задача сокращается на половину элементов массива. Если ключ меньше, чем средний элемент массива, то дальнейший поиск осуществляется в первой половине массива, а если больше, то во второй половине. Поиск продолжается до тех пор, пока ключевой элемент не станет равным среднему элементу или пока оставшийся подмассив содержит хотя бы один элемент, не равный ключевому.

Отметим, что если для определения места включения очередного элемента в упорядоченную часть массива для метода сортировки вставками использовать двоичный поиск, то удастся увеличить скорость сортировки.

Приведённые ниже задачи позволяют ознакомиться с простыми методами поиска и сортировки.

#### 2.4.1. Реализовать алгоритм линейного поиска с барьером.

Реализовать поиск заданного элемента в упорядоченном массиве методом бинарного поиска.

```
program birpoisk;
const n=9;
type IArray=array [1..n] of integer;
var l,r,k:byte;
    x:integer;
    A:IArray;
begin
    writeln('Введите элементы массива');
    for k:=1 to n do readln(A[k]);
    writeln('Введите искомый элемент:');
    readln(x);
    l:=1; r:=n+1;
    while l<r do
    begin
        k:=(l+r) div 2;
        if A[k]<x then l:=k+1
        else r:=k
    end;
    if A[r]=x then writeln('Это',r:1,
    '-й элемент массива.')
    else writeln('Ключевой элемент не найден.')
end.
```

2.4.2. Удалить из упорядоченного массива элементы, равные заданному значению.

2.4.3. Вставить в упорядоченный массив элемент, имеющий заданное значение, если он отсутствует в массиве.

2.4.4. Даны два упорядоченных по возрастанию массива. Получить путём слияния исходных массивов новый упорядоченный

- по неубыванию;
- по возрастанию.

2.4.5. Упорядочить массив по неубыванию, используя метод прямого выбора.

2.4.6. Упорядочить массив по невозрастанию методом пузырька.

2.4.7. Упорядочить массив по неубыванию методом двоичных вставок, получая упорядоченный массив

- на месте старого массива;
- в другом массиве.

2.4.8. Даны прямоугольная матрица. Упорядочить строки матрицы:

- по неубыванию сумм элементов строк;
- по неубыванию наименьших элементов строк;
- по невозрастанию наибольших элементов строк.

## 2.5 СТРОКИ

Примечание: во всех приведённых задачах под текстом понимается строка или группа строк, например, содержимое текстового файла. Группы символов, разделённые пробелами и не содержащие пробелов внутри себя, будем называть словами.

Напишите программу, печатающую заданную последовательность из десяти строк, упорядочив их по алфавиту.

```
program alfstr;
const n=10;
type SArray=array [1..n] of string;
var A:SArray;
    i,j,l:integer;
begin
    writeln('Введите строки.');
    for i:=1 to n do readln (A[i]); (*чтение массива строк*)
    for i:=1 to n do
```

```

begin
    l:=1;
    for j:=1 to n do
        (*определим номер непустой строки, *)
        if (A[j]<A[i]) and (A[j]<>'') then l:=j;
        (*которую нужно вывести на печать*)
        writeln(A[l]);
        A[l]:=' *напечатанную строку уничтожаем*
    end
end.

```

2.5.1. Напишите программу, которая принимает на входе текст, содержащий последовательность заглавных и строчных букв, затем печатает этот текст только литерами заглавных букв.

2.5.2. В данном тексте

- а) найти наибольшее количество цифр, идущих в нём подряд;
- б) определить наличие символов, отличных от букв и пробела;
- в) заменить буквы N, D, и Y соответственно на H, D, и Y.

2.5.3. Для каждого символа данного текста указать, сколько раз он встречается в тексте.

2.5.4. Выяснить, является ли данный текст десятичной записью целого числа.

2.5.5. Напечатать в алфавитном порядке все различные русские буквы, входящие в данный текст.

2.5.6. Для данного текста проделайте следующие действия:

- а) для каждого из слов укажите, сколько раз оно встречается среди слов текста;
- б) найдите слова, начинающиеся и оканчивающиеся одной и той же буквой;
- в) найдите самое длинное слово;
- г) укажите слова, в которых доля гласных максимальна;
- д) удалите заданное слово;
- е) удалите слова, встречающие более одного раза.

2.5.7. Проверьте, имеется ли в заданном тексте баланс открывающих и закрывающих скобок.

2.5.8. Палиндром – слово или текст, одинаково читающийся слева направо и справа налево. Выясните, является ли заданный текст палиндромом.

2.5.9. Зашифруйте текст, циклически сдвигая его вправо или влево.

30

2.5.10. Даны два текста. Для них

- а) найдите множество всех слов, встречающихся в каждом тексте;
- б) найдите самое длинное общее слово обоих тестов;
- в) найдите самое короткое из слов первого текста, которого нет во втором.

2.5.11. Замените в данном тексте каждую из групп стоящих рядом точек, одной точкой.

2.5.12. Замените в тексте каждую точку троеточием.

2.5.13. Если в данном тексте нет символа «\*», то оставьте его без изменений, иначе каждую из букв, предшествующих вхождению символа «\*», заменить символом «/».

2.5.14. Если в данном тексте есть буквы, то каждый из символов, следующих за первой группой букв, замените точками. В противном случае текст оставьте без изменений.

2.5.15. Найдите количество предложений в заданном тексте.

## 2.6. ЗАПИСИ

2.6.1. Таблица содержит сведения: шифр, фамилию и инициалы сталевара, вес (в кг.) запланированных и израсходованных им в течение месяца материалов. Получить список сталеваров, сэкономивших материалы.

2.6.2. Ввести список студентов, содержащий следующую информацию о каждом: фамилию, имя, отчество, номер курса, номер группы, сведения о проживании (дома, в общежитии или на квартире). Получить список студентов, проживающих в общежитии.

2.6.3. Ведомость содержит сведения за месяц о пропусках занятий студентам: фамилию и инициалы, номер группы, количество часов, пропущенных по уважительной и неуважительной причине. Получить список студентов, пропустивших более двух часов по неуважительной причине.

2.6.4. Анкета содержит сведения об аттестации студентов группы: фамилию и инициалы, номер группы, аттестацию (0 или 1) по каждому из шести предметов. Получить список неаттестованных студентов.

2.6.5. Ведомость содержит итоги сессии: фамилию и инициалы, номер курса, номер группы, оценки полученные в сессию. Получить список студентов, претендующих на повышенную стипендию.

2.6.6. Даны сведения об использовании машинного времени ВЦ

31

кафедрами: название кафедры, расход машинного времени: по плану и фактически. Получить список кафедр, превышающих плановый расход машинного времени.

2.6.7. Анкета содержит сведения в виде: фамилия, имя, отчество, величина зарплаты, стаж работы, количество детей. Получить список сотрудников, стаж которых превышает 10 лет, а зарплата ниже 700 рублей.

2.6.8. Таблица содержит сведения о перевозках авиапассажиров на рейсах аэропорта: номер рейса, маршрут, марка самолёта, общие затраты на рейс, количество пассажиров. Подсчитать среднюю стоимость перевозки одного пассажира на рейсе, а также итоговые сведения по затратам и количеству пассажиров и среднюю стоимость перевозки одного пассажира по аэропорту. Результаты представить на экране в виде таблиц.

2.6.9. Даны сведения о расходе топлива на автобазах города: № автобазы, количество израсходованного топлива, количество автомашин на базе. Подсчитать средний расход топлива на одну автомашину на каждой базе и в целом по городу. Результат оформить в виде таблицы.

2.6.10. Даны записи о проданных товарах: наименование, артикул, количество, цена за единицу. По каждой записи получить сумму выручки за данный вид товара. Выдать общую сумму реализации товаров.

2.6.11. Задана следующая информация: фамилия и инициалы рабочего, выработка (в %) и количество прогулов за месяц. Получить списки прогульщиков и работников, не выполнивших план.

2.6.12. Таблица поставщиков содержит сведения: предприятие - отправитель грузов, плановое и фактически полученное число контейнеров. Получить список поставщиков, не выполнивших плановые обязательства.

2.6.13. Данна ведомость на зарплату сотрудников подразделения. Получить общую сумму, приходящуюся на подразделение; указать минимальную и максимальную зарплату сотрудников.

2.6.14. Расписание движения пригородных поездов Ростов-Азов содержит номер поезда, время отправления из Ростова и время прибытия в Азов. Выдать сведения о поездах, отправляющихся из Ростова не раньше 7.30 и прибывающих в Азов не позднее 16.00.

2.6.15. Даны записи о расходовании электроэнергии на заводах отрасли: номер завода, Ф.И.О. директора, и главного энергетика, количество израсходованной (в тыс. кВт·ч) энергии и запланированный

32

расход. Получить по каждому заводу величину отклонения фактического расхода от запланированного. Подсчитать суммарные значения для отрасли в целом.

2.6.16. Даны сведения за отчётный период о материальных ценностях в стоимостном выражении по филиалам завода: номер и название филиала, наличие ценностей на начало периода (в руб.), суммы полученных ценностей, сумма выбывших ценностей. Получить ведомость, содержащую: стоимость ценностей по каждому филиалу на конец периода, а также итоговые сведения по всему заводу в целом.

2.6.17. Таблица содержит сведения о времени выполнения заданий на ЭВМ (время в секундах с точностью до сотых долей): шифр задания, Ф.И.О. программиста, общее время прохождения задания, время центрального процессора. Получить процент процессорного времени по каждому заданию, а также суммы по видам времени по всем заданиям и средний процент времени центрального процессора по всем заданиям.

2.6.18. Ведомость содержит сведения о работниках предприятия: табельный номер, фамилия и инициалы, количество отработанных часов за месяц, стоимость одного часа работы и аванс. Требуется

а) вычислить сумму к выдаче и выдать результаты в виде таблицы с указанием табельного номера, фамилии, инициалов и суммы;  
б) осуществить поиск записей с заданными табельными номерами с выдачей фамилии и инициалов работника и полученного аванса.

2.6.19. Ведомость содержит сведения о сотрудниках предприятия: фамилию и инициалы, даты рождения, поступления на работу и поступления на данное предприятие (все даты задаются в виде ГГММЧЧ), разряд и количество детей. Получить списки сотрудников, имеющих:

а) разряд не ниже третьего и возраст не более 23 лет;  
б) общий стаж работы не менее 20 лет и не достигших возраста 50 лет;  
в) детей и стаж работы на данном предприятии не менее 5 лет;  
г) общий стаж работы не менее 20 лет и стаж работы на данном предприятии свыше 10 лет.

2.6.20. Багаж пассажира характеризуется количеством вещей и общим весом вещей. Ведомость содержит сведения о багаже нескольких пассажиров. Требуется

а) найти багаж, средний вес одной вещи в котором отличается не более, чем на 0.3 кг от общего среднего веса одной вещи;

б) найти число пассажиров, имеющих более двух вещей и число пассажиров, количество вещей которых превосходит среднее число вещей;

в) выяснить, имеется ли пассажир, багаж которого состоит из одной вещи весом менее 30 кг.

## 2.7 МНОЖЕСТВА

*Описать функцию, подсчитывающую количество элементов в множестве A, состоящем из целых чисел от 0 до 99.*

```
type ISet=set of 0..99; (*множество целых чисел от 0 до 99*)
function kol(A:ISet):integer;
var i:byte;j:integer;
begin
  j:=0;
  for i:=0 to 99 do if i in A then j:=j+1;
  (*сколько чисел из интервала [0;99] входят в A*)
  kol:=j;
end;
```

2.7.1. Дано сто целых чисел от 1 до 50. Напишите программу, определяющую, сколько из них являются числами Фибоначчи и сколько чисел, первая значащая цифра в записи которых 1 или 2.

2.7.2. Дано множество символов от «а» до «z». Напишите программу, выводящую элементы этого множества на печать в алфавитном порядке.

2.7.3. Дан текст из цифр и строчных букв латинского алфавита, за которым следует точка. Определить, каких букв – гласных (а, е, и, о, у) или согласных – больше в этом тексте.

2.7.4. Дано множество символов А и символ х. Напишите программу, строящую множество В из множества А по следующему правилу:

- добавлением элемента х;
- удалением элемента х.

2.7.5. Дан текст из строчных латинских букв, за которым следует точка. Напечатайте:

- первые вхождения букв в текст, сохраняя при этом их исходный взаимный порядок;
- все буквы, входящие в текст не менее двух раз;

в) все буквы, входящие в текст по одному разу.

2.7.6. Дан текст, за которым следует точка. В алфавитном порядке напечатать по одному разу все строчные русские гласные буквы, входящие в этот текст.

2.7.7. В возрастающем порядке напечатать все целые числа из диапазона 1...10000, представимые в виде  $n^2 + m^2$ , где  $n, m \geq 0$ .

2.7.8. В порядке убывания напечатать все целые числа из диапазона 1...4900, которые представимы в виде  $n^2 + 2k^2$ , но не представимы в виде  $7ij + j + 3$ ,  $n, k, i, j \geq 0$ .

2.7.9. Дано целое n от 2 до 1000. Используя метод решета Эратосфена, напечатайте в убывающем порядке все простые числа из диапазона  $n..2n$ .

Суть этого метода: выписываются все целые числа, большие единицы; выбирается первое из них (это 2 – простое число) и вычёркиваются все кратные ему числа, кроме него самого; затем берётся следующее из невычёркнутых чисел (это 3 – также простое число) и вычёркиваются все кратные ему, кроме него самого; действия повторяются для всех невычёркнутых ранее чисел. В конце концов останутся только простые числа, начиная с двух.

2.7.10. Предположим, что некоторые магазины торгуют следующими товарами: хлеб, масло, молоко, мясо, рыба, соль, сыр, колбаса, сахар, чай, кофе. Ассортимент магазина – это множество товаров из этого набора. Компания владеет двадцатью магазинами. По информации об ассортименте, представленном в каждом из этих магазинов построить множество: А – множество продуктов, которые есть во всех магазинах; В – множество продуктов, каждый из которых есть хотя бы в одном магазине; С – множество продуктов, которых нет ни в одном магазине.

2.7.11. Перечислены имена одноклассников: Вася, Володя, Ира, Лиза, Марина, Миша, Наташа, Олег, Оля, Света, Юля. В гости к каждому из них приглашается некоторое подмножество ребят. Создайте массив, содержащий сведения о приглашенных и, используя его, определите

а) есть ли среди них хотя бы один человек, побывавший в гостях у всех одноклассников;

б) есть ли люди, не приглашённые ни к одному из одноклассников.

2.7.12. Данна непустая последовательность слов из строчных русских букв; между соседними словами – запятая, за последним словом точка. Напечатать в алфавитном порядке:

- а) все гласные буквы, которые входят в каждое слово;  
 б) все согласные буквы, которые не входят ни в одно слово;  
 в) все звонкие согласные буквы, которые входят хотя бы в одно слово;  
 г) все глухие согласные, которые не входят хотя бы в одно слово;  
 д) все согласные буквы, которые входят только в одно слово;  
 е) все глухие согласные, которые не входят только в одно слово;  
 ж) все звонкие согласные, которые входят более чем в одно слово;  
 з) все гласные буквы, которые не входят более чем в одно слово;  
 и) все звонкие согласные буквы, которые входят в каждое нечетное слово и не входят ни в одно четное слово;  
 к) все глухие согласные буквы, которые входят в каждое нечетное слово и не входят хотя бы в одно четное слово.

2.7.13. Дан список городов некоторого региона: а, б, с, д, е, ф, г, х. Для каждого из них известно множество городов, в которые можно попасть из данного города за один автобусный рейс без пересадки. Напишите программу, которая, используя массив, содержащий такую информацию для всех городов, будет определять

- а) множество городов, в которые можно попасть из заданного пользователем города;  
 б) кратчайший (в смысле количества пересадок) путь между парой заданных городов.

2.7.14. Напишите программу, реализующую игру «быки и коровы»: компьютер загадывает четырёхзначное число, не содержащее двух одинаковых цифр; пользователь пытается назвать цифры, из которых состоит это число; на каждом шаге сообщается количество «быков» – правильно отгаданных – и «коров» – не правильно отгаданных цифр; игра продолжается, пока не будет набрано четыре «быка».

## 2.8. СПИСКИ

**При меч ани е:** для решения большинства задач этого раздела могут быть использованы рекурсивные алгоритмы.

*Следующая программа размещает в динамической памяти массив Arr вещественных чисел и распечатывает все положительные элементы массива.*

```
var i,j:integer;
    Arr:array[1..1000] of ^real;
begin
```

```
    for i:=1 to 1000 do new(Arr[i]);
    (*размещение массива в динамической памяти*)
    for i:=1 to 1000 do
        if Arr[i]^>0 then Write(Arr[i]^);
    (* обращение к элементам массива по их адресам *)
    end.
```

2.8.1. - Разместить в динамической памяти массив вещественных чисел, используя тип pointer, встроенные функции типа word – seg(x) и ofs(x), возвращающие сегментную часть адреса и смещение и функцию ptr(seg,ofs:word): pointer, которая создает значение указателя, совместимое с указателями любого типа. Распечатать все элементы массива, меньшие двух.

2.8.2. В динамической памяти разместить массив записей следующей структуры: наименование товара, количество на складе, цена единицы продукции, дата поступления.

- а) напечатать все товары, поступившие до 1999 года.  
 б) найти товар, количество которого на складе максимально.

2.8.3. Переписать массив из предыдущего примера в другое место динамической памяти, оставив в массиве только те товары, количество которых на складе меньше 100 единиц.

2.8.4. Написать программу, позволяющую из массива записей (пример 2.8.2) составить массив записей следующей структуры: наименование товара, количество на складе, и разместить его в динамической области памяти.

2.8.5. Разместить в динамической памяти массив целых чисел. Определить наибольшее из чисел.

2.8.6. Разместить в динамической памяти массив целых чисел.  
 а) ссылаются ли разные элементы массива на одно и то же место динамической памяти?

б) ссылаются ли разные элементы массива на равные между собой числа?

2.8.7. Дан массив ссылок на вещественные числа. Написать программу для

а) нахождения наибольшего из чисел, на которые ссылаются элементы этого массива;

б) нахождения первого из элементов массива, ссылающегося на отрицательное число (результат считать равным nil, если такого элемента нет);

в) проверки, есть ли в массиве хотя бы два элемента, ссылающихся на одинаковые числа;

г) преобразования заданного массива по следующему правилу: все элементы, ссылающиеся на одинаковые числа, заменяются на первый из этих элементов.

2.8.8. Используя представление текста, как массива ссылок на строки одинаковой длины (ссылки на пустые строки равны nil, если текст не пуст, то в начале массива нет ссылок, равных nil), описать:

а) процедуру, считывающую последовательность литер до первой точки и формирующую из них текст, дополняя его при необходимости пробелами;

б) функцию для подсчёта числа непустых строк в данном тексте;

в) функцию, проверяющую, есть ли в тексте строка с номером  $i$  и, если есть, печатающую её  $j$ -й элемент;

г) процедуру, меняющую местами  $i$ -ю и  $j$ -ю строки текста;

д) процедуру, заменяющую  $i$ -ю строку на копию  $j$ -й строки;

е) процедуру, добавляющую после  $j$ -й строки копию  $j$ -й;

ж) процедуру, удаляющую из текста  $j$ -ю строку;

з) функцию, определяющую, входит ли литера  $c$  в текст  $i$ , если входит, присваивающую переменным  $i$  и  $j$  «координаты» первого вхождения;

и) процедуру, печатающую построчно весь текст.

2.8.9. Напишите программу, которая помещает 25 случайных целых чисел в диапазоне от 0 до 100 в упорядоченный список. Вычислите сумму и среднее арифметическое элементов этого списка.

2.8.10. Описать функцию или процедуру, которая:

а) определяет, является ли список пустым;

б) находит среднее арифметическое элементов непустого списка;

в) заменяет все вхождения заданного элемента другим заданным;

г) переносит в конец непустого списка его первый элемент;

д) переносит в начало непустого списка его последний элемент;

е) меняет местами первый и последний элементы непустого списка.

2.8.11. Напишите программу вывода элементов списка на печать в обратном порядке, использующую рекурсивный и нерекурсивный алгоритмы.

2.8.12. Написать программу, в результате работы которой заданный во входном файле текст будет распечатан в обратном порядке. Для решения задачи использовать список.

2.8.13. Описать процедуру, которая по исходному списку строит два новых списка: один из положительных элементов, а другой из остальных элементов списка.

2.8.14. Опишите процедуры, которые вставляют:

- а) в начало списка заданное количество элементов;
- б) в конец списка заданное количество элементов;
- в) новый элемент после первого элемента непустого списка;
- г) новый элемент перед последним элементом списка, состоящего, по крайней мере, из двух элементов;

д) за каждым вхождением заданного элемента другой заданный элемент;

е) перед первым вхождением заданного элемента (если такое существует) другой заданный элемент;

ж) в упорядоченный список новый элемент так, чтобы не нарушить его упорядоченности.

2.8.15. Опишите процедуру, которая удаляет

а) из непустого списка первый элемент;

б) из списка второй элемент, если такой есть;

в) из списка за каждым вхождением заданного элемента один элемент, если такой есть и он отличен от заданного;

г) из списка все подряд идущие повторяющиеся элементы, кроме первого из них;

д) из списка все повторяющиеся элементы, кроме их первых вхождений;

е) из списка все отрицательные элементы;

ж) из непустого списка последний элемент.

2.8.16. Напишите программу, которая создаёт список из 10 символьных элементов, а затем создаёт второй список, содержащий копию первого списка, но в обратном порядке.

2.8.17. Дано целое число  $n > 1$ , за которым следует  $n$  вещественных чисел. Напечатать эти числа в порядке их неубывания.

2.8.18. Описать процедуру или функцию, которая:

а) проверяет на равенство два заданных списка;

б) определяет, входит ли один заданный список в другой;

в) проверяет, есть ли в заданном списке хотя бы два одинаковых элемента;

г) добавляет в конец одного заданного списка все элементы другого списка;

д) добавляет в заданный список за первым вхождением данного элемента (если оно есть) все элементы другого списка;

е) изменяет в списке все ссылки так, чтобы его элементы оказались расположеными в обратном порядке.

2.8.19. Напишите программу, объединяющую два упорядоченных по неубыванию списка в один упорядоченный по неубыванию список:

а) построив новый список;

б) меняя соответствующим образом ссылки в исходных списках.

2.8.20. Опишите процедуру, которая формирует список, включив в него по одному разу элементы, которые:

а) входят хотя бы в один из двух заданных списков;

б) входят одновременно в оба заданных списка;

в) входят в один из списков, но не входят в другой.

2.8.21. Даны три списка. В первом списке все вхождения второго списка (если такие есть) замените на третий.

2.8.22. Во входном файле записана последовательность символов. Описать процедуру, переписывающую содержимое этого файла в другой в алфавитном порядке.

2.8.23. Используя списки, определите, симметричен ли текст, записанный во входном файле.

2.8.24. Данна последовательность из не менее чем двух различных натуральных чисел, за которой следует 0. Напечатать в обратном порядке все числа, находящиеся между наибольшим и наименьшим членами этой последовательности.

2.8.25. Дано слово, состоящее из букв русского алфавита. Представить его в виде линейного связного списка, каждое звено которого содержит одну букву и ссылку на следующую букву. Преобразовать слово по одному из правил:

а) из слова удалить все последующие вхождения первой его буквы;

б) после каждой буквы исходного слова вставить букву А;

в) в каждой очередной паре букв поменять буквы местами;

г) каждое вхождение в слово первой его буквы заменить словом ДА;

д) если последняя буква входит в слово несколько раз, то оставить только последнее вхождение этой буквы;

е) в слове оставить только первое вхождение каждой из букв, удалив остальные.

2.8.26. Используя представление слов в виде списков (см. предыдущую задачу), создайте список, элементами которого являются слова. Опишите процедуру, которая:

а) в этом списке переставляет местами первое и последнее непустое слово;

б) печатает текст, состоящий из первых (последних) букв всех непустых слов построенного списка;

в) удаляет из непустых слов списка их первые (последние) буквы;

г) определяет количество слов в построенном списке, отличных от первого (последнего) слова;

д) определяет количество слов в построенном списке, равных первому (последнему) слову.

2.8.27. Из массива, записанного в динамической памяти (пример 2.8.2), создать односвязный список.

2.8.28. Задать десять чисел, поместить их в список. Удалить из списка все отрицательные числа.

2.8.29. Построить список, состоящий из вещественных чисел. Подсчитать среднее арифметическое элементов списка.

2.8.30. Текст, заданный в виде строкового массива, записать в список. Заменить в списке группу подряд идущих пробелов на один пробел.

2.8.31. Текст, заданный в виде строкового массива, записать в список. Одно слово от другого в тексте отделяется пробелом. Распечатать список, причем каждое слово текста печатать с новой строчки.

2.8.32. Сделать реализацию динамического массива на базе линейного односвязного списка.

2.8.33. Сделать реализацию матрицы на базе линейного односвязного списка. Информационная часть элементов списка содержит три значения: номер строки, номер столбца, значение элемента матрицы.

2.8.34. Построить список, содержащий сведения о людях: фамилия, место рождения, год рождения, место работы;

а) если в списке присутствует Сидоров 1977 года рождения из Ростова, то перед ним вставить Попова 1950 года рождения из Казани;

б) подсчитать количество элементов в списке.

2.8.35. Построить список сведений о семьях: фамилия, количество членов семьи, количество детей;

а) добавить элемент в начало (конец) списка;

б) добавить элемент в середину списка.

2.8.36. Построить список сведений о мебели: вид мебели, название, цена;

а) удалить из списка все кухонные шкафы;

б) если в списке встречаются подряд два одинаковых вида мебели, то удалить первый из этих элементов.

2.8.37. Построить список сведений о писателях: фамилия, год рождения, год смерти, название самого известного произведения;

а) проверить, есть ли среди элементов списка Короленко В.Г. (1853–1921), «Дети подземелья».

б) переставить местами первый и последний элементы списка.

2.8.38. Построить список сведений о кошках: название породы, кличка, возраст, пол;

а) распечатать данные всех кошек женского пола;

б) распечатать кличку самой молодой кошки;

в) распечатать название породы и клички всех кошек мужского пола в возрасте от года до двух.

2.8.39. Создать два списка и проверить их на равенство. Является ли один из списков подсписком другого?

2.8.40. Создать два списка. Сформировать третий список, состоящий из элементов, принадлежащих как первому, так и второму спискам.

2.8.41. Используя список, определить, симметричен ли заданный в виде строки текст.

2.8.42. Циклический список – это список, у которого последний элемент ссылается на первый элемент списка. Используя циклический список, зашифровать заданный текст, задавая сдвиг алфавита на  $k$  позиций вправо (влево).

2.8.43. В списке заданы сведения об успеваемости студентов: название специальности, курс, номер группы, средний балл, код студента. Записать в стек студентов, упорядочивая их таким образом, чтобы на вершине стека был студент с наилучшей успеваемостью, в конце стека – студент с худшей успеваемостью. Напечатать фамилии лучших по успеваемости студентов.

## 2.9. СТЕКИ

Стек является частным случаем однонаправленного списка, в котором новые узлы могут добавляться в стек и выталкиваться из него только на вершине. Для представления стека создаётся однонаправленный список, в котором элементы стека располагаются в обратном порядке. Однако для облегчения понимания механизмов работы со структурой данных «стек» на первых этапах можно использовать представление стека в виде массива, в начале которого располагаются элементы стека. При этом запоминается номер последнего элемента стека. Это представление не вполне соответствует пониманию стека как динамической структуры данных.

Задать десять целых чисел, поместить их в список. Распечатать все числа, большие единицы.

42

```
Type uc=^ch;
  ch=record (*тип элемента списка*)
    inf:integer;
    link:uc;
  end;
var first,pr,tr:uc;
  i,k:integer;
begin
  for i:=1 to 10 do
begin
  writeln('Введите',i,'ое число');
  read(k);
  if i=1 then (*creation первого элемента списка*)
    begin
      new(first); (*адрес начала списка хранится в переменной
first*)
      first^.inf:=k;
      tr:=first;
    end
  else
    begin
      new(pr); (*creation очередного элемента списка*)
      pr^.inf:=k;
      tr^.link:=pr;
      tr:=pr;
    end
  end;
  pr^.link:=nil; (*ввод чисел закончен, список создан*)
  tr:=first;
  while tr^.link do
begin
  if tr^.inf<1 then write(tr^.inf); (*печать очередного
элемента списка*)
  tr:=tr^.link; (*переход к следующему элементу списка*)
end;
end.
```

2.9.1. Используя описанное выше представление стека, описать процедуру или функцию:

- проверяющую, является ли стек пустым;
- создающую пустой стек;

- в) добавляющую заданный элемент в стек;  
 г) удаляющую элемент из стека, присваивая его значение выходному параметру.

2.9.2. Данна экзаменационная ведомость с оценками по  $m$  предметам. Напечатать списки отличников, задолжников и всех остальных студентов, для накопления фамилий по разным спискам организовать три стека.

2.9.3. Постфиксной формой записи выражения  $a\Delta b$  (или его обратной польской нотацией) называется запись, в которой знак операции размещён за операндами:  $ab\Delta$ . Для вычисления значения выражения, записанного в постфиксной форме, применяют следующий алгоритм. Выражение просматривается слева направо. Если встречается операнд, то его значение заносится в стек, а если встречается знак операции, то из стека извлекаются два последних элемента (операнда это операции), над ними выполняется эта операция и результат помещается в стек. По окончании просмотра в стеке останется только одно число – значение выражения. Опишите процедуру, вычисляющую значение выражения, записанного во входном текстовом файле в обратной польской нотации.

2.9.4. Для преобразования инфиксной формы выражения в постфиксную (см. задачу 2.9.3) можно использовать следующий алгоритм. Выражение просматривается слева направо. Операнды добавляются в стек  $X$ , а левые скобки и операции – в стек  $Y$ . Встретив правую скобку, отыскиваем в стеке соответствующую ей левую. При этом всё, что сверху – выталкивается из стека  $Y$  и вталкивается в стек  $X$ . Очередная операция помещается в стек  $Y$ , если только лежащая ниже операция имеет более низкий приоритет или оказывается левой скобкой. В противном случае, вначале элементы из стека  $Y$  выталкиваются в  $X$  до тех пор, пока не встретится левая скобка, дно стека или операция с более низким приоритетом, а потом очередная операция помещается в  $Y$ . Исходное выражение обрабатывается до тех пор, пока не встретится знак равенства.

Оператор	Приоритет
*	3
/	3
+	2
-	2
(	1
=	0

Если считать этот знак операцией с самым низким приоритетом, то в результате последнего выталкивания в стеке  $X$  окажется записанным исходное выражение в постфиксной форме. В приведённой таблице приоритеты операциям присвоены так, чтобы избежать проверки на равенство очередной операции левой скобке. Напишите процедуру, переписывающую выражение в инфиксной форме из входного текстового файла в выражение в постфиксной форме, сохраняя результат в выходном текстовом файле.

2.9.5. Опишите процедуру, переводящую выражение, записанное в постфиксной форме, в инфиксную форму.

2.9.6. Напишите программу, которая вычисляет постфиксное выражение (полагая, что оно правильное), состоящее из цифр и операций. Модифицируйте эту программу так, чтобы она могла обрабатывать операнды целого типа, значения которых больше 9.

2.9.7. Напишите программу,читывающую некоторое арифметическое выражение, преобразующее его по правилам обратной польской записи и вычисляющее его значение для вводимых пользователем значений входящих в него переменных.

2.9.8. Напишите программу, которая печатает построчно заданный во входном файле текст, располагая символы в строках в обратном порядке.

2.9.9. Используя стек, определите, является ли строка палиндромом (т.е. строкой, которая одинаково читается как в прямом, так и в обратном направлениях). Программа должна игнорировать пробелы и знаки пунктуации в строке.

2.9.10. Сделать реализацию ограниченного стека со сведениями о студентах, в котором студенты упорядочены по убыванию успеваемости. В вершине стека находится студент с худшей успеваемостью. Ограниченный стек предполагает наличие дополнительной операции: проверить, есть ли свободное место в стеке? Ограниченный стек не может вместить больше некоторого фиксированного числа элементов. Предусмотреть операцию отчисления пяти студентов с худшей успеваемостью, если в стеке не осталось свободного места.

2.9.11. Используя стек, проверить правильность арифметического выражения.

а) проверить число открывающих и закрывающих круглых скобок в выражении;

б) проверить, следует ли перед и за знаком арифметической операции число.

## 2.10. ОЧЕРЕДИ

Очередь – это частный случай списка. Элементы в очередь добавляются только в её конец (или хвост), а удаляются только из начала (или головы). Для представления очереди используется однодirectionalnyy список и запоминаются указатели на его начало и конец. Однако для облегчения восприятия материала можно воспользоваться представлением очереди в виде массива, в котором элементы очереди занимают группу соседних компонент, индексы первой и последней из которых запоминаются; при этом, когда очередь достигает правого края массива либо все её элементы сдвигаются к левому краю, либо новые элементы записываются в начало массива.

2.10.1. Используя описанные выше способы представления очереди, опишите процедуру или функцию, которая:

- а) создаёт пустую очередь;
- б) проверяет, является ли очередь пустой;
- в) добавляет в конец очереди заданный элемент;
- г) удаляет из очереди первый элемент, запоминая его значение в параметре  $x$ .

2.10.2. Постройте очередь со сведениями о квартирах: район, где расположена квартира, количество комнат, метраж, этаж, цена;

- а) добавить элемент в очередь;
- б) удалить элемент из очереди;
- в) проверить, пуста ли очередь;
- г) распечатать очередь;
- д) сделать очередь пустой.

2.10.3. Дан файл действительных чисел. Выведите на печать вначале все числа, меньшие заданного  $a$ , затем – все числа из отрезка  $[a, b]$ , и, наконец, – все остальные числа, сохраняя исходный взаимный порядок в каждой из этих трёх групп чисел.

2.10.4. Дан текстовый файл. Перепишите его содержимое в другой файл, перенося при этом в конец каждой строки все входящие в неё цифры с сохранением исходного взаимного порядка среди цифр и среди остальных литер строки.

2.10.5. Дано множество имён: Анна, Мария, ..., Пётр. Матрица, состоящая из булевых элементов, строится по следующему правилу: элемент в строке  $x$  и столбце  $y$  является истиной, если человек по имени  $x$  является родителем человека по имени  $y$ . Напишите программу, которая по заданному имени формирует выходной файл, за-

писывая в него сначала всех детей человека, имеющего это имя, затем всех внуков, затем правнуков и т.д.

2.10.6. Напишите программу, моделирующую линию ожидания ёмкостью 10 человек, если для обслуживания клиента и удаления его из очереди требуется 2 единицы времени. Вероятность появления одного нового клиента в течение единицы времени равна  $p$ , где  $0 < p < 1$ . Предполагается, что в течение единицы времени никогда не появляются два и более клиентов. Проследите за переполнением очереди как за функцией от  $p$ .

2.10.7. Напишите программу, которая моделирует очередь в супермаркете. Покупатели появляются в ней случайным образом в интервале от 1 до 4 минут (рассматриваются только целые значения). Обслуживается очередной покупатель в очереди тоже случайным образом в интервале от 1 до 4 минут (также рассматриваются только целые значения). Запустите модель супермаркета при условии 12-часового рабочего дня (720 минут), используя следующий алгоритм:

- а) определите момент появления первого покупателя, сгенерировав случайное число в диапазоне от 1 до 4;
- б) в момент появления первого покупателя определите время его обслуживания, спланируйте появление следующего покупателя (добавьте к текущему времени случайное целое в диапазоне от 1 до 4) и начните обслуживание первого покупателя;
- в) в каждую минуту дня, если появится следующий покупатель, поставьте его в очередь и спланируйте время появления следующего, если обслужен очередной покупатель, то исключите из очереди следующего покупателя и определите время его обслуживания.

2.10.8. Используя условие предыдущей задачи запустите моделирование супермаркета в течение 720 минут и определите:

- а) максимальное число покупателей в очереди;
- б) максимальное время ожидания покупателей;
- в) поведение модели в случае, если интервалы времени изменить с 1–4 минут до 1–3 минут.

2.10.9. Используя понятие очереди, упорядочить абитуриентов по дате подачи документов. Сведения об абитуриентах заданы в виде: фамилия, год рождения, средний балл аттестата, место проживания. Предусмотреть процедуры печати фамилии абитуриента, сдавшего документы первым, добавления абитуриента в очередь, проверки, пуста ли очередь.

## 2.11. ДВУСВЯЗНЫЕ СПИСКИ

Двусвязным является список, каждый элемент которого имеет два указателя. Оба под списка двусвязного списка должны иметь свои указатели начала односвязных списков.

Построить двусвязный список всех целых чисел от 0 до 20. Второй указатель в списке указывает на следующее четное число. Распечатать все четные числа, большие десяти.

```
Type uc1=^ch1;
    ch1=record (*тип элемента списка*)
        inf:integer;
        link1,link2:uc1;
    (*link1 указывает на следующий элемент в первом подсписке,
    link2 – во втором*)
    end;
var first1,first2,pr,tr:uc1;
i,k:integer;
begin
(*creation первого подсписка*)
for i:=0 to 20 do
begin
writeln('Введите',i,'е число');
read(k);
if i=0 then
begin
new(first1);
(*адрес начала первого подсписка хранится в переменной
first1*)
first1^.inf:=k;
tr:=first1;
end
else
begin
new(pr);
(*creation очередного элемента первого подсписка*)
pr^.inf:=k;
tr^.link1:=pr;
tr^.link2:=nil; (*второй подсписок пока пуст*)
tr:=pr;
end
end;
tr^.link1:=nil;
tr^.link2:=nil; (*ввод чисел закончен, первый подсписок со-
здан*)
(*creation второго подсписка*)
first2:=first1; (*первые элементы обоих подсписков совпа-
дают*)
if first1<>nil then tr:=first1^.link1
else tr:=first1;
pr:=first2;
while tr<>nil do
begin
if tr^.inf mod 2 = 0 then
begin
(*установление связи между соседними элементами второго под-
списка*)
pr^.link2:=tr;
pr:=tr;
end;
tr:=tr^.link1;
end;
tr:=first2;
while tr<>nil do
begin
if tr^.inf>10 then write(tr^.inf);
(*печать очередного элемента списка*)
tr:=tr^.link2; (*переход к следующему элементу списка*)
end;
end.
```

```
end;
```

2.11.1. Построить двусвязный список со сведениями о спортивных командах: название, количество очков, занимаемое место, место-расположение команды. Первый указатель указывает на следующую команду в списке, второй – на следующую команду из города Ростова;

- напечатать названия команд из Ростова, набравших 10 очков;
- проверить, есть ли среди элементов списка команда Спартак, 20 очков, 1 место, город Москва.

2.11.2. Построить двусвязный список сведений о высших учебных заведениях: название, город, количество студентов. Второй ука-

затель указывает на следующий ростовский вуз. Напечатать название ростовского вуза с наибольшим количеством студентов.

2.11.3. Проверить, совпадают ли два двусвязных списка, то есть, состоят ли оба списка из одинаковых элементов.

2.11.4. Создать двусвязный список. Подсчитать количество элементов в каждом из его подсписков.

2.11.5. Проверить, совпадают ли подсписки двусвязного списка.

2.11.6. Осуществить реализацию динамического массива на базе двусвязного списка.

2.11.7. Проверить, используя двусвязный список, является ли заданный текст палиндромом.

2.11.8. Задать десять целых чисел. Записать их в двусвязный список, первый указатель которого указывает на следующее целое число, второй – на следующее положительной числа. Добавить в список число «2».

2.11.9. Построить двусвязный список сведений о товарах: название, цена, размер. Второй указатель списка указывает на следующий товар с ценой, меньшей 10 у.е.

а) если в списке есть костюм 40-ого размера с ценой 50 у.е., то перед этим товаром в списке вставить платье 34 размера ценой 40 у.е.;

б) если в списке есть туфли 36-ого размера с ценой 5 у.е., то перед этим товаром вставить чулки 8 размера ценой 4 у.е.

2.11.10. Построить двусвязный список сведений о компьютерах: название, цена, размер оперативной памяти, быстродействие. Второй указатель списка указывает на следующий компьютер с ценой, большей 1000 у.е. Если в списке встречаются подряд два элемента с одинаковым объемом оперативной памяти, то удалить первый из этих элементов.

2.11.11. Построить двусвязный список сведений о продуктах: название, цена, место изготовления. Второй указатель указывает на следующий ростовский продукт. Если в списке встретились продукты «ростовская колбаса по 34 руб.» и «ростовское молоко по 5 руб.», то поменять эти товары местами в списке.

2.11.12. Построить двусвязный список сведений о городах: название, численность населения, страна. Второй указатель указывает на следующий город России. Добавить в список город Ростов с миллионом жителей из России.

2.11.13. Построить двусвязный список сведений о животных: название, количество, место обитания. Второй указатель указывает на

следующее животное из Африки. Удалить из списка всех животных из Африки, которых осталось от 10 до 100 экземпляров.

2.11.14. Построить двусвязный список сведений о собаках: название породы, возраст, пол. Второй указатель указывает на следующую собаку возрастом от года до двух. Добавить в список бульдога 5-ти лет, кобеля.

2.11.15. Построить двусвязный список сведений о реках: название, протяженность, месторасположение. Второй указатель указывает на следующую реку с протяженностью равной 1000 км. Удалить из списка все реки Ростовской области.

2.11.16. Построить двусвязный список сведений о странах: название, численность населения. Второй указатель указывает на следующую страну с численностью населения меньше, чем миллион жителей. Добавить в список Монако, 50 тысяч жителей.

2.11.17. Задать текст в виде строкового массива. Признак конца текста – точка, одно слово от другого отделяется пробелом. Поместить текст в двусвязный список, элементами которого являются слова, первый указатель списка указывает на очередное слово, второй – на очередное слово четной длины. Распечатать все слова нечетной длины.

2.11.18. Построить двусвязный список одномерных массивов, первый указатель которого указывает на следующий массив, второй – на следующий массив, сумма элементов которого больше 100. Удалить из списка все массивы с суммой элементов, равной 177.

2.11.19. Построить двусвязный список символов, первый указатель которого указывает на следующий символ, второй – на следующий символ, не являющийся буквой. Если в списке есть подряд две буквы «А», то между ними вставить букву «Б».

2.11.20. Построить двусвязный список вещественных чисел, второй указатель которого указывает на следующее вещественное число, сумма цифр которого равна 10. Удалить из списка числа, сумма цифр которых равна 5.

2.11.21. Построить двусвязный список, распечатать все его элементы, входящие в один подсписок, и не входящие в другой.

2.11.22. Даны два односвязных списка. Построить двусвязный список, состоящий из всех элементов обоих односвязных списков, в котором второй указатель указывает на следующий элемент, входящий в оба односвязных списка. Распечатать все элементы второго подсписка.

## 2.12. ДЕРЕВЬЯ

Бинарное или двоичное дерево есть конечное множество узлов, которое или пусто, или состоит из корня и двух непересекающихся бинарных деревьев, называемых левым и правым поддеревом данного дерева.

Построить двоичное дерево и распечатать его. Использовать три варианта обхода: префиксный или прямой – корень, левое поддерево, правое; постфиксный или концевой – левое поддерево, правое, корень; и инфиксный или обратный – левое поддерево, корень, правое.

Пусть вершины дерева задаются буквами. Признак конца текста – точка. Пустые поддеревья обозначаются звездочками, то есть задается вид дерева.

Пример вводимого текста – СТУ\*\*\*ДЕ\*Н\*\*Г\*\*. При работе с деревом используются рекурсивные алгоритмы.

```
'Type uk2=^tree;
tree=record (*тип элемента дерева*)
inf:char;
ltree,rtree:uk2;
end;
var root:uk2;
s:string;
i:integer;
procedure create(var link:root); (*создание дерева*)
var pt:uk2;
begin
i:=i+1;
if s[i]<>'. ' then
if s[i]<>'*' then
begin
new(pt);
pt^.inf:=s[i];
link:=pt;
create(link^.ltree);
create(link^.rtree);
end
else
link:=nil;
end;
```

ИК2

ЧК2

```
procedure prefix(link:^uk2); (*префиксный обход*)
begin
if link<>nil then
begin
writeln(link^.inf);
prefix(link^.ltree);
prefix(link^.rtree);
end;
end; INFIX
procedure infix(link:uk2); (*инфиксный обход*)
begin
if link<>nil then
begin
prefix(link^.ltree);
writeln(link^.inf);
prefix(link^.rtree);
end;
end; POSTFIX
procedure postfix(link:uk2); (*постфиксный обход*)
begin
if link<>nil then
begin
prefix(link^.ltree);
prefix(link^.rtree);
writeln(link^.inf);
end;
end;
```

2.12.1. Используя стек, построить двоичное дерево нерекурсивным способом.

2.12.2. Записать текст в двоичное дерево. Одно слово от другого отделяется в тексте пробелом (одним или несколькими).

- a) заменить все слова «машина» на «снег»;
- б) заменить все слова «да» на «весна».

2.12.3. Используя дерево, распечатать текст, заданный в виде строки, в обратном порядке.

2.12.4. Используя дерево, определить в тексте слово максимальной длины. Одно слово от другого отделяется в тексте пробелом.

2.12.5. Используя дерево, подсчитать количество слов в тексте. Одно слово от другого отделяется пробелом.

2.12.6. Используя дерево, удалить из текста все слова «студент». Одно слово от другого отделяется пробелом.

2.12.7. Используя дерево, определить, встречается ли в тексте слово «зима». Одно слово от другого отделяется пробелом.

2.12.8. Построить дерево. Подсчитать число его вершин.

2.12.9. Построить бинарное дерево. Записать в другое бинарное дерево все повторяющиеся элементы первого дерева.

2.12.10. Используя бинарное дерево и различные варианты его обхода, зашифровать некоторый текст.

2.12.11. Построить бинарное дерево. Проверить, совпадают ли все левые и правые поддеревья этого дерева.

2.12.12. Построить два бинарных дерева. Проверить, содержат ли эти деревья одинаковые элементы, используя

- а) рекурсивные процедуры;
- б) не используя рекурсии.

2.12.13. Дерево называется равновесным, если число узлов у правого и левого поддеревьев каждой вершины разнится не более, чем на единицу. Построить равновесное дерево и распечатать его.

2.12.14. Проверить, является ли заданное бинарное дерево равновесным.

2.12.15. Удалить из равновесного дерева все буквы «A».

2.12.16. Добавить в равновесное дерево заданный элемент, не нарушая равновесности дерева.

2.12.17. Построить равновесное дерево. Если в дереве встречаются две подряд буквы «O», то вставить между ними букву «B», не нарушая равновесность дерева.

2.12.18. Построить бинарное дерево. Переписать все элементы этого дерева в равновесное бинарное дерево.

2.12.19. Уровень корня дерева равен нулю. Уровень любого другого узла всегда на единицу больше, чем уровень узла, ссылающегося на данный.

54

2.12.20. Задать дерево и определить уровни узлов дерева, содержащих букву «а».

2.12.21. Глубиной дерева называется максимальная величина уровней его узлов. Построить дерево и определить его глубину.

2.12.22. Используя стек или очередь, напечатать все элементы дерева по уровням: сначала корень, потом вершины первого уровня и т.д.

2.12.23. Создать дерево. Найти в дереве длину (число ветвей) пути от корня до ближайшей вершины с заданным элементом.

2.12.24. Деревом сортirovki называется двоичное дерево, построенное по следующему правилу. Если элемент меньше, чем корень, то он должен добавляться в левое поддерево, если больше – то в правое. Если в дереве есть корень с таким же значением, то добавления не происходит. Построить обычное дерево, элементами которого являются числа. Удалить из дерева все повторяющиеся элементы.

2.12.25. Вершина дерева называется терминальной или листом, если она не ссылается ни на один элемент дерева. Распечатать все терминальные вершины дерева сортirovki.

2.12.26. Если вершина дерева не является листом, то она называется нетерминальной. Построить дерево сортirovki и распечатать все его нетерминальные вершины, большие заданного числа.

2.12.27. Построить дерево сортirovki. Если в дереве встречаются числа «3», «4», «5», то удалить число «4» из дерева.

2.12.28. Добавить в дерево сортirovki число «5», если его в дереве нет.

2.12.29. Построить дерево сортirovki и исключить из дерева число «2».

2.12.30. Построить два дерева сортirovki. Проверить, равны ли суммы элементов этих деревьев.

2.12.31. Найти минимальный элемент дерева сортirovki.

2.12.32. Проверить содержит ли дерево сортirovki заданный элемент.

2.12.33. Подсчитать число вершин дерева сортirovki, больших заданного числа.

## 2.13. ТИПИЗИРОВАННЫЕ ФАЙЛЫ

Записать все целые числа от 1 до 1000 в файл.

```
var f:file of integer;
```

```

i:integer;
begin
assign(f,'ttt.dat'); (*установка связи между файловой
переменной и файлом*)
rewrite(f); (*открытие нового файла для записи*)
for i:=1 to 1000 do
write(f,i); (*запись элемента в файл*)
close(f); (*закрытие файла*)
end.

```

2.13.1. Создать файл целых чисел.

- подсчитать число элементов файла, больших единицы.
- подсчитать общее число элементов файла.

2.13.2. Найти наименьший элемент файла целых чисел и распечатать его.

2.13.3. Проверить упорядочены ли элементы файла целых чисел по возрастанию.

2.13.4. Напечатать третий с конца элемент файла целых чисел.

2.13.5. Напечатать элементы файла целых чисел, образующие возрастающую последовательность.

2.13.6. Упорядочить элементы файла целых чисел по возрастанию.

2.13.7. Создать два файла вещественных чисел. Проверить, совпадает ли содержимое этих файлов.

2.13.8. Дан массив вещественных чисел. Все числа, большие единицы, записать в один файл, меньшие единицы --- в другой, равные -- в третий. Распечатать все три файла.

*Добавить элемент в конец файла целых чисел.*

```

var f:file of integer;
i:integer;
begin
assign(f,'ttt.dat');
(*установка связи между файловой переменной и файлом*)
reset(f); (*открытие файла для добавления в него числа*)
seek(f,filesize(f));
(*указатель файла установлен на конец файла*)
writeln('введите добавляемое число');
readln(i);
write(f,i); (*запись добавляемого элемента в файл*)
close(f); (*закрытие файла*)
end.

```

*Добавить элемент в начало файла целых чисел. Информация из исходного файла f переносится во временный файл copia, в котором на первое место помещено добавляемое число. Затем файл f стирается, файл copia переименовывается в f.*

```

var f,copia:file of integer;
i:integer;
begin
assign(f,'ttt.dat');
reset(f);
(*открытие файла для чтения информации из него*)
assign(copia,'prom.dat');
rewrite(g); (*открытие файла для записи*)
writeln('введите добавляемое число');
readln(i);
write(copia,i); (*запись добавляемого элемента*)
while not eof(f) do
begin
read(f,i);
write(copia,i); (*копирование файла f в copia*)
end;
close(f); close(copia); (*закрытие файлов*)
erase(f); (*стирание файла f*)
rename(copia,'ttt.dat'); (*переименование файла*)
end.

```

2.13.9. Создать файл целых чисел G. Переписать содержимое файла G в файл F в обратном порядке.

2.13.10. Если файл целых чисел имеет нечетную длину, то поменять местами вторую и предпоследнюю компоненты файла.

2.13.11. В файле целых чисел перед каждым положительным числом вставить число ноль.

2.13.12. В файле целых чисел, если перед положительным числом стоит отрицательное, то удалить это отрицательное число.

2.13.13. Заданы два файла целых чисел. Уничтожить тот файл, сумма элементов которого больше.

2.13.14. Переставить местами первую и последнюю компоненты файла целых чисел.

2.13.15. Создать файл сведений о телевизорах: марка, цена, диагональ, страна-изготовитель:

а) напечатать все сведения о телевизорах с ценой 250 у.е.,  
б) напечатать все сведения о телевизорах марки «Рубин» с ценой  
250 у.е.

2.13.16. Создать файл сведений о служащих: фамилия, оклад, возраст, должность. Определить служащего с минимальным окладом. Увеличить его оклад на 20 у.е.

2.13.17. Создать файл сведений о кинофильмах: название, жанр, продолжительность. Проверить, есть ли среди элементов файла кинокартина «Гараж», комедия, продолжительность 80 мин.

2.13.18. Создать файл сведений о костюмах F: название, цена, размер. Удалить из файла все праздничные костюмы ценой 200 у.е. 52 размера.

2.13.19. Создать файл сведений о художественных картинах G: автор, год написания, размер. Если в файле подряд расположены две картины одного автора, то удалить первую из них.

2.13.20. Создать файл сведений о заводах: название, количество работающих, название производимой продукции, месторасположение. Если в файле присутствует завод «Ростсельмаш» с численностью 10 тысяч человек, производящий комбайны и расположенный в городе Ростове-на-Дону, то перед ним вставить завод «ЗИЛ» с численностью 30 тысяч человек, производящий автомашины и расположенный в городе Москва.

2.13.21. Создать файл целых чисел F. Все положительные числа из файла F записать в файл G, все отрицательные числа – в файл H.

2.13.22. Создать два файла вещественных чисел. Записать в третий файл сначала все элементы первого файла, затем – второго.

2.13.23. Создать два файла вещественных чисел. Записать в третий файл на  $i$ -ое место минимальный из  $i$ -ых элементов первого и второго файлов.

2.13.24. Создать файл целых чисел F. Записать в файл G все числа из F, которые являются полными квадратами.

2.13.25. Задать два файла целых чисел. Записать в третий файл все числа первого файла, которых нет во втором.

2.13.26. Создать файл, элементами которого являются массивы целых чисел. Найти максимальный элемент каждого массива и распечатать массив с максимальным из максимальных элементов.

## 2.14. ТЕКСТОВЫЕ ФАЙЛЫ

Текстовые файлы — это файлы, компонентами которых являются строки. Для текстовых файлов сохраняются основные операции, применяемые для типизированных файлов, и добавляется ряд новых операций.

Распечатать содержимое текстового файла. Каждую строку файла печатать с новой строкой.

```
var f:text;
    s:string;
    i:integer;
begin
  write('введите имя файла');
  readln(s);
  assign(f,s);
  (*установка связи между файловой переменной и файлом*)
  reset(f); (*открытие файла*)
  while not eof(f) do
    begin
      readln(f,s); (*чтение строки из файла*)
      writeln(s); (*печать строки*)
    end;
  close(f); (*закрытие файла*)
end.
```

2.14.1. Создать текстовый файл. Подсчитать количество строк в нем.

2.14.2. Дан файл сведений о мотоциклах F: марка, цена, количество лошадиных сил.

2.14.3. Создать и распечатать текстовый файл, содержащий марки всех мотоциклов из файла F.

2.14.4. Дан файл сведений о людях G: фамилия, возраст, пол. Создать два текстовых файла, один из которых содержит фамилии всех женщин из файла G, второй – всех мужчин. Распечатать оба построенных текстовых файла.

2.14.5. Найти длину максимальной строки в текстовом файле.

2.14.6. Удалить из текстового файла строку максимальной длины.

2.14.7. Напечатать построчно содержимое текстового файла.

2.14.8. Подсчитать количество пустых строк в текстовом файле.

2.14.9. Создать текстовый файл. Одно слово от другого в каждой строке файла отделяется пробелом (одним или несколькими). Подсчитать количество слов в строке наибольшей длины.

2.14.10. Проверить, есть ли в текстовом файле слово «студент». Одно слово от другого отделяется пробелом.

2.14.11. Если в какой-то строчке текстового файла встречается слово «студент», то в начало этой строки вставить слово «РГУ». Одно слово от другого отделяется пробелом.

2.14.12. Все слова «школьник» в текстовом файле заменить словом «abituriyent».

2.14.13. Одно слово от другого отделяется пробелом.

2.14.14. Создать текстовый файл. Переписать его содержимое в другой текстовый файл.

2.14.15. В текстовом файле найти строку с максимальным количеством слов и распечатать ее. Одно слово от другого отделяется пробелом.

2.14.16. Создать текстовый файл, одно слово от другого в котором отделяется пробелом. Найти строки, в которых пять слов, и удалить в этих строчках два последних слова.

2.14.17. Если в текстовом файле есть строчка длины 10 символов, то заменить ее пустой строкой.

2.14.18. Удалить из текстового файла все строчки, в которых встречается два слова «университет».

2.14.19. В текстовом файле перед каждой пустой строкой вставить строку «Наступил учебный год».

2.14.20. В текстовом файле в начало каждой пустой строки вставить слова «Пустая строка».

2.14.21. Переписать содержимое одного текстового файла в другой, без пустых строчек.

2.14.22. Переписать содержимое одного текстового файла в другой, удваивая все пустые строчки.

2.14.23. Создать текстовый файл. Зашифровать строчку наибольшей длины этого файла, используя коды букв. Зашифрованный текст записать в файл целых чисел.

2.14.24. Задан файл целых чисел, каждое из которых представляется собой код буквы. Расшифровать текст и записать его в текстовый файл.

2.14.25. Написать программу, которая преформирует текстовый файл, разбивая его на строки. Каждая строка или заканчивается «», или содержит ровно 30 букв.

2.14.26. Задать текстовый файл. Распечатать его построчно, вставляя в начало каждой строки ее номер.

2.14.27. Если текстовый файл имеет десять компонент, то удалить из файла три его первые компоненты.

## 2.15. НЕТИПИЗИРОВАННЫЕ ФАЙЛЫ

Скопировать файл f в файл g.

```
var f,g:file;
    s:string;
    bufer:array[1..100] of char;
    k1,kolv:integer;
begin
  write('введите имя исходного файла');
  readln(s);
  assign(f,s);
  (* установка связи между файловой переменной и файлом*)
  reset(f,1); (*размер блока при чтении установлен равным 1*)
  write('введите имя выходного файла');
  readln(s);
  assign(g,s);
  rewrite(g,1); (*размер блока при записи установлен равным 1*)
  repeat
    (*чтение информации из файла в переменную buffer*)
    blockread(f,bufer,sizeof(buffer),kolv);
    (*запись информации в файл из переменной buffer*)
    blockwrite(g,bufer,kolv,k1);
  until (kolv=0) or (k1<>kolv);
  close(f);
  close(g);
end.
```

2.15.1. Создать нетипизированный файл, состоящий из элементов разной структуры: записей, массивов, целых чисел и строк и распечатать его.

2.15.2. Определить длину нетипизированного файла.

2.15.3. Удалить вторую компоненту нетипизированного файла.

2.15.4. Создать нетипизированный файл. Добавить в начало файла массив записей структуры: название книги, автор, цена, год издания.

2.15.5. Создать нетипизированный файл. Добавить в конец файла заданное целое число.

2.15.6. Создать нетипизированный файл. Вставить между второй и третьей компонентами файла массив вещественных чисел из двадцати элементов.

2.15.7. Переписать информацию из одного нетипизированного файла в другой, увеличив в два раза длину записи файла.

2.15.8. Переписать информацию из одного нетипизированного файла в другой, уменьшив в два раза длину записи файла.

2.15.9. Дано два типизированных файла: файл целых чисел и текстовый файл. Записать из них информацию в нетипизированный файл.

2.15.10. Распечатать нечетные компоненты нетипизированного файла.

## 2.16. СОРТИРОВКА ФАЙЛОВ

Сортировка обменами основана на обмене элементов с индексами  $i$  и  $j$  таких, что  $i < j$ , но  $i$ -ый элемент больше  $j$ -ого. Просмотр элементов продолжается до тех пор, пока обмен происходит. Метод, основанный на обмене соседних элементов, называется методом пузырька.

Упорядочить заданный файл целых чисел по возрастанию методом пузырька.

```
var f:file of integer;
    i,j:integer;
    Q:boolean;
begin
  assign(f,'ttt.dat');
  (*установка связи между файловой переменной и файлом*)
  reset(f); (*открытие файла для обработки*)
  repeat
    Q:=true;
    while not eof(f) do
      begin
        read(f,i);
        read(f,j);
        if i>j then
          begin
            (*перемещение указателя файла на две позиции назад от текущего
            положения*)
            seek(f,filepos(f)-2);
            write(f,j); (*запись в файл*)
            write(f,i);
            Q:=false;
          end;
      end;
  until Q;
  close(f); (*закрытие файла*)
end.
```

```
seek(f,filepos(f)-2);
write(f,j); (*запись в файл*)
write(f,i);
Q:=false;
      end;
    end;
  seek(f,1); (*указатель файла установлен на начало*)
until Q;
close(f); (*закрытие файла*)
end.
```

2.16.1. Создать файл записей со сведениями о писателях: фамилия, год рождения, год смерти, название самого известного произведения. Упорядочить писателей в файле по возрасту методом пузырька.

2.16.2. Создать файл записей со сведениями о телевизорах: марка, цена, диагональ, страна-изготовитель. Упорядочить телевизоры по цене методом пузырька.

2.16.3. Метод сортировки обменами, основанный на сравнении пар элементов, расположенных далеко друг от друга, называется методом быстрой сортировки. Выберем произвольный элемент массива. Будем просматривать массив слева, пока не найдем элемент меньший или равный выбранному, затем просмотрим массив справа, пока не встретим элемент, больший выбранного. Поменяем найденные элементы местами и продолжим просмотр, пока не сойдемся в середине массива. В результате массив разделится на две части: левую, с элементами меньшими выбранного, и правую, с элементами, большими выбранного. К полученным двум частям массива применяют тот же процесс до тех пор, пока каждая часть не будет состоять из одного элемента.

Создать файл записей со сведениями о кошках: название породы, кличка, возраст, пол. Упорядочить кошек в файле по возрасту методом быстрой сортировки, используя рекурсивный вариант алгоритма.

2.16.4. Создать файл записей со сведениями о товарах: название, цена, размер. Упорядочить товары по цене методом быстрой сортировки, используя нерекурсивный вариант алгоритма.

2.16.5. Создать файл записей со сведениями о костюмах: название, цена, размер. Упорядочить костюмы по цене методом быстрой сортировки, используя рекурсивный вариант алгоритма.

2.16.6. На методе обмена основан метод параллельной сортиров-

ки. Он предполагает, что  $N$  сортируемых элементов помещены в  $N$  процессоров. В параллельных алгоритмах процессоры разбиваются на группы, и шаг алгоритма выполняется одновременно для всех групп.

Создать файл записей со сведениями о компьютерах: название, цена, размер оперативной памяти, быстродействие. Упорядочить компьютеры по цене, используя параллельный алгоритм сортировки.

2.16.7. Сортировка включением заключается в следующем: пусть упорядочено некоторое множество элементов, тогда очередной элемент включаем в отсортированную последовательность, то есть, устанавливаем его на место среди других упорядоченных элементов так, чтобы не нарушилась упорядоченность.

2.16.8. Создать файл записей со сведениями о людях: фамилия, место рождения, год рождения, место работы. Упорядочить людей в файле по возрасту методом простого включения.

2.16.9. Создать файл записей со сведениями о служащих: фамилия, оклад, возраст, должность. Упорядочить служащих согласно их окладу методом простого включения.

2.16.10. Метод простого включения, в котором поиск в упорядоченном файле основан на сравнении очередного элемента со средним элементом в уже отсортированной последовательности (результат сравнения позволяет определить в какой половине продолжать поиск; к выбранной части файла применяется тот же метод), называется методом простого включения с использованием бинарного потока.

Создать файл записей со сведениями о семьях: фамилия, количество членов семьи, количество детей. Упорядочить семьи в файле по количеству членов в семьях методом простого включения с использованием бинарного потока.

2.16.11. В методе Шелла (методе сортировки включением с убывающим шагом) вместо систематического включения элемента с индексом  $j$  в подпоследовательность предшествующих ему элементов файла – включают этот элемент в подсписок, содержащий элементы с индексами  $j-h, j-2h, j-3h$  и так далее, где  $h=\text{const}>0$ . Таким образом, формируется файл, в котором серии элементов, отстоящих друг от друга на шаг  $h$ , сортируются отдельно. Затем процесс возобновляется с новым шагом  $h_1 < h$  и так далее. Серии сортируются с шагами  $h, h_1, \dots, h_n$ , причем  $h_n=1$ , что обеспечивает окончательную сортировку файла.

Создать файл записей со сведениями о мебели: вид мебели, название, цена. Упорядочить мебель в файле по цене методом Шелла.

2.16.12. Создать файл записей со сведениями о странах: название, численность населения. Упорядочить страны по численности населения методом Шелла.

2.16.13. При сортировке выбором ищется минимальный (максимальный) элемент, он меняется местами с первым элементом. Затем то же самое выполняется с оставшимися элементами файла (файл просматривается, начиная со второго элемента и так далее).

Создать файл записей со сведениями о продуктах: название, цена, место изготовления. Упорядочить продукты в файле по цене методом выделения минимального элемента.

2.16.14. Создать файл записей со сведениями о реках: название, протяженность, месторасположение. Упорядочить реки по их протяженности методом выделения максимального элемента.

2.16.15. Разновидностью сортировки посредством выбора является сортировка с помощью двоичного дерева, корнем которого является наименьший элемент файла. Наименьший элемент исключается из дерева. Элемент, передвинувшийся в корень, вновь будет наименьшим и его тоже можно исключить из дерева. После  $N$  таких шагов дерево станет пустым и процесс сортировки заканчивается.

Создать файл записей со сведениями о городах: название, численность населения, страна. Упорядочить города по численности населения с помощью двоичного дерева.

2.16.16. Создать файл записей со сведениями о кинофильмах: название, жанр, продолжительность. Упорядочить кинофильмы по их продолжительности с помощью двоичного дерева.

2.16.17. Под сортировкой слиянием понимают объединение двух или более упорядоченных файлов в один упорядоченный файл. Сравниваются наименьшие элементы обоих файлов, минимальный из них записывается в третий файл. Эти действия повторяются до тех пор, пока есть элементы в исходных двух файлах.

Создать два файла записей со сведениями о животных: название, количество, место обитания. Записать информацию из этих двух файлов в третий файл, упорядочивая животных по их численности.

2.16.18. Создать два файла записей со сведениями о заводах: название, количество работающих, название производимой продукции, месторасположение. Записать информацию из этих двух файлов в третий файл, упорядочивая заводы по количеству работающих.

2.16.19. Операции обмена медленнее операций сравнения. Поз-

тому в некоторых случаях прибегают к созданию дополнительного файла, называемого индексным. Индексный файл содержит ключ (поле), по которому идет сортировка и номер элемента с таким ключом в исходном файле. Сортируется не сам файл, а индексный файл.

Создать файл записей со сведениями о собаках: название породы, возраст, пол. Упорядочить собак по возрасту с использованием файла индексов.

2.16.20. Создать файл записей со сведениями о художественных картинах: автор, год написания, размер. Упорядочить картины по времени написания с использованием файла индексов.

## 2.17. РЕКУРСИЯ

Используя рекурсивную функцию, найти  $n!$ :

```
var i:integer;
function n!(k:integer):integer;
begin
  if (k=0) or (k=1) then n!:=1
  (*выход из рекурсивной функции*)
  else n!:=k*n!(k-1)
  (*рекурсия - функция вызывает себя же, но от другого значения
параметра*)
  end;
begin
writeln('задайте неотрицательное целое число');
read(i);
writeln('факториал числа', i, '=', n!(i));
end.
```

2.17.1. Описать рекурсивную функцию, которая методом деления отрезка пополам находит корни заданной функции на отрезке.

2.17.2. Используя рекурсивную функцию, найти  $n$ -ый член последовательности  $x_n = 3x_{n-1}$ ,  $x_0 = 1$ .

2.17.3. Используя рекурсивную функцию, найти сумму первых 40 членов последовательности из предыдущего примера.

2.17.4. Используя рекурсивную функцию, найти максимальный из первых 50-и членов последовательности  $x_n = 5x_{n-1} + 1$ ,  $x_0 = 0$ .

2.17.5. Используя рекурсивную функцию, определить, сходится ли последовательность  $x_n = (5x_{n-1} - 2) / 10$ ,  $x_0 = 5$ .

2.17.6. Используя рекурсивную функцию, распечатать первые 10 членов последовательности  $x_n = 5x_{n-1} - 20$ ,  $x_0 = 2$ , которые больше 10.

2.17.7. Используя рекурсивную функцию, распечатать номера первых 10-и членов  $x_n = 2x_{n-3} + 3x_{n-1} - 4$ ,  $x_0 = 2$ ,  $x_1 = 0$ ,  $x_2 = 1$  последовательности, которые больше 15.

2.17.8. Используя рекурсивную функцию, найти сумму целых чисел, кратных трем от 100 до 200.

2.17.9. Используя рекурсивную функцию, найти произведение целых чисел, кратных пяти от 61 до 121.

2.17.10. Используя рекурсивные функции, установить, что больше: двадцатый член последовательности  $x_n = 2x_{n-1} - 10$ ,  $x_0 = 1$  или сумма целых чисел, кратных семи от 100 до 200.

2.17.11. Используя рекурсивные функции, установить, что больше: десятый член последовательности  $x_n = 5x_{n-1} + 7$ ,  $x_0 = 3$  или произведение целых чисел, кратных двадцати одному от 100 до 200.

2.17.12. Используя рекурсивные функции, перевести число из десятичной системы исчисления в восьмеричную и распечатать его.

2.17.13. Используя рекурсивные функции, перевести число из двоичной системы исчисления в десятичную и распечатать его.

2.17.14. Используя рекурсивные функции, напечатать все простые числа от 1 до  $n$ .

2.17.15. Используя рекурсивные функции, подсчитать

$$\sum_{i=1}^n \frac{1}{i^2 + 4}$$

2.17.16. Используя рекурсивные функции, подсчитать

$$\prod_{i=5}^{n-1} \frac{10}{i^3 + 10}$$

2.17.17. Используя рекурсивные функции, установить, сходится ли ряд

$$\sum_{i=3}^{\infty} \frac{i}{i^4 - 10}$$

## 2.18. ИСПОЛЬЗОВАНИЕ СТАНДАРТНЫХ И СОБСТВЕННЫХ МОДУЛЕЙ ПРИ СОЗДАНИИ ПРОГРАММНЫХ КОМПЛЕКСОВ

Используя стандартный модуль crt, провести очистку экрана перед выполнением программы.

```
uses crt;
(*подключение библиотеки стандартных модулей crt*)
begin
clrscr; (*вызов процедуры очистки экрана*)
...
end.
```

2.18.1. Используя стандартный модуль crt, заставить динамик компьютера звучать с различной частотой.

2.18.2. Используя стандартный модуль graph, установить на экране несколько окон различных размеров и залить их разным цветом.

2.18.3. Используя стандартный модуль graph, нарисовать несколько семи- и девятиугольников различных размеров.

2.18.4. Используя стандартный модуль graph, нарисовать квадраты и треугольники различных размеров.

2.18.5. Используя стандартный модуль graph, нарисовать окружности и эллипсы и закрасить их.

2.18.6. Используя стандартный модуль graph, сохранить и затем выдать некоторое изображение.

2.18.7. Используя стандартный модуль dos, распечатать текущую дату и текущее время.

2.18.8. Используя стандартный модуль dos, установить текущую дату и текущее время.

2.18.9. Используя стандартный модуль dos, распечатать число свободных байт на всех имеющихся дисках.

2.18.10. Используя стандартный модуль dos, распечатать адрес прерывания и его содержимое.

2.18.11. Написать программу вычисления площадей треугольника, прямоугольника, параллелограмма, круга в виде набора процедур. Поместить эти процедуры в разные модули. Описания глобальных переменных

- а) оставить в основном модуле;
- б) вынести в отдельный модуль описаний;
- в) внести в каждый из созданных модулей.

## ГЛАВА 3 ГРАФИКА И АНИМАЦИЯ

3.1.1. Изобразить на экране точку, пересекающую с постоянной скоростью экран справа налево параллельно его горизонтальной оси.

3.1.2. К условию предыдущей задачи добавляется следующее требование: как только точка доходит до левого края, в этот момент от правого края в строке, выбранной с помощью датчика случайных чисел, начинает свое движение другая точка и т.д. Цвет точки также может выбираться с помощью датчика случайных чисел.

3.1.3. Усложним условие предыдущей задачи: очередная точка начинает движение от правого края экрана несколько раньше, чем предыдущая точка доходит до левого края. Попытаться добиться того, чтобы одновременно на экране двигались три, четыре точки.

3.1.4. Изобразить на экране точку, движущуюся по окружности с постоянной угловой скоростью.

3.1.5. В ряде следующих задач речь идет о движении точки. Вместо точки на экране можно использовать небольшой круг, прямоугольник или изображение предмета: волана для бадминтона, стрелы, самолета и т. д. Движение может сопровождаться звуком.

3.1.6. Изобразить на экране прямую, вращающуюся в плоскости экрана вокруг одной из своих точек. Задачу можно усложнить дополнительным требованием, чтобы цвет прямой изменялся при переходе от предыдущего положения к следующему.

3.1.7. Изобразить на экране отрезок, вращающийся в плоскости экрана вокруг

- а) своей середины;
- б) своего конца;
- в) точки, делящей отрезок в отношении 1:3.

3.1.8. Усложним условие двух предыдущих задач: в плоскости экрана должны вращаться, каждая вокруг своей точки, две прямые.

3.1.9. В условие предыдущей задачи вносится дополнение: должна быть предоставлена возможность управления с клавиатуры расстоянием между центрами вращения.

3.1.10. Изобразить на экране две движущиеся точки, траектории которых являются концентрическими окружностями. Угловая ск

рость точки, движущейся по внутренней окружности, должна быть несколько меньше, чем угловая скорость точки, движущейся по внешней окружности (обе скорости – постоянные величины). При этом

а) обе точки вращаются в одном направлении (например, по часовой стрелке);

б) точки вращаются в противоположных направлениях.

3.1.11. Изобразить одновременное вращение двух стрелок – большой и малой, при котором одному полному обороту большой стрелки соответствует  $1/12$  оборота малой стрелки (как на циферблате часов). Стрелки можно для простоты заменить отрезками.

3.1.12. Изобразить на экране правильный треугольник, вращающийся в плоскости экрана вокруг своего центра.

3.1.13. Изобразить на экране разносторонний треугольник, вращающийся в плоскости экрана вокруг своего центра тяжести.

3.1.14. Изобразить на экране прямоугольник, вращающийся в плоскости экрана вокруг своего центра.

3.1.15. Изобразить на экране прямоугольник, вращающийся в плоскости экрана вокруг одной из своих вершин.

3.1.16. Условие этой задачи отличается от предыдущей тем, что требуется сохранять на экране все высвеченные положения геометрической фигуры. Задачу можно еще усложнить дополнительным требованием, чтобы цвет фигуры изменялся при переходе от предыдущего положения к следующему.

3.1.17. Изобразить движущуюся прямую, которая в каждый момент касается окружности данного радиуса, центр которой совпадает с центром экрана. Точка касания перемещается по окружности с постоянной угловой скоростью. Сама окружность невидима.

3.1.18. Изобразить равнобедренный треугольник, вращающийся с постоянной угловой скоростью вокруг своей высоты, расположенной параллельно вертикальной оси экрана.

3.1.19. Изобразить на экране движение шара по биллиарду без луз.

3.1.20. В условие предыдущей задачи вносится дополнение: шар должен оставлять за собой светящийся след.

3.1.21. Изобразить на экране точку, пересекающую экран равнозакономерно в вертикальном направлении.

3.1.22. Изобразить на экране приближающийся издали шар. По какому закону возрастает видимый диаметр шара с течением времени?

3.1.23. Получить мультфильм «Круги на воде», используя семь концентрических окружностей. Центры окружностей должны быть совмещены с центром экрана, а радиусы изменяться от 40 до 82 пиксел, увеличиваясь на 7 пикселя с каждой следующей окружностью. Иллюзия движения должна создаваться последовательной сменой цветов всех окружностей, начиная с внутренней и кончая внешней. Процесс смены цветов следует повторить не менее десяти раз.

3.1.24. Получить на экране два слова ТЕСТ, движущиеся по экрану по одной горизонтальной прямой навстречу друг другу. Первое слово должно двигаться от левого края экрана к правому, второе — от правого края к левому. Движение должно выполняться до полного совмещения слов.

3.1.25. В рисованных мультфильмах иллюзия движения создается последовательной сменой кадров, каждый из которых фиксирует очередное положение движущегося объекта. Используя этот принцип, получить мультфильм, показывающий:

- а) идущего человечка;
- б) бегущего человечка;
- в) человечка, выполняющего приседания;
- г) человечка, выполняющего сигнализацию флагом.

3.1.26. Для построения отдельных кадров мультфильма воспользоваться фигурками, описанными в рассказе А. Конан-Дойля «Пляшущие человечки».

3.1.27. Аналогично предыдущей задаче получить спортивный мультфильм:

- а) о метании диска;
- б) о беге с барьерами;
- в) о подтягивании на перекладине;
- г) о прыжках в длину;
- д) о гребле;
- е) о поднятии штанги.

Построение отдельных кадров выполнить на основе олимпийской символики.

3.1.28. Получить звуковой мультфильм «Человечек, танцующий под музыку». Танец может заключаться в выполнении самых простых движений, например, притоптываний ногой или приседаний, в такт мелодии, воспроизводимой по звукогенератору.

3.1.29. Аналогично предыдущей задаче получить звуковой мультфильм «Человечек, играющий на гитаре». Игра на гитаре может изображаться, например, с помощью перемещения правой руки

по струнам гитары вверх и вниз. Мультфильм должен сопровождаться воспроизведением популярной мелодии.

3.1.30. Дано 15 чисел  $a_1, b_1, c_1, a_2, b_2, c_2, \dots, a_i, b_i, c_i$ . Каждая тройка чисел  $a_i, b_i, c_i$  ( $i=1, \dots, 5$ ) определяет параболу  $y = ax^2 + bx + c$ . Построить графики всех этих парабол на отрезке  $[-5, 5]$ . График параболы с номером  $i$  должен иметь цвет с номером  $i \bmod 3 + 1$ .

3.1.31. Написать программу, в ходе выполнения которой одновременно с отысканием приближенного решения уравнения  $f(x)=0$  с помощью алгоритма хорд строится иллюстрация применения этого алгоритма. График функции  $f(x)$ , хорды и вспомогательные вертикальные отрезки должны быть окрашены в разные цвета.

3.1.32. В ходе приближенного вычисления площади криволинейной трапеции получить на экране иллюстрацию применения соответствующего алгоритма.

3.1.33. Метательное устройство находится на расстоянии 800 м от цели. Изменяя начальную скорость полета снаряда  $V_n$  и угол  $\alpha$ , под которым он выпускается, можно регулировать дальность полета. Написать программу с исходными данными  $V_n$  и  $\alpha$ . В левом нижнем углу экрана должен появляться прямоугольник, изображающий метательный аппарат, в правом нижнем углу – отрезок, изображающий цель. Точка, изображающая снаряд, должна переместиться по экрану от аппарата в сторону цели по параболе определяемой исходными данными. В случае попадания снаряда в цель, отрезок, изображающий цель, исчезнет.

3.1.34. Написать игровую программу, позволяющую выполнять несколько попыток при «стрельбе» по цели. В случае попадания в цель она должна менять свое положение на экране. После окончания «стрельбы» должна сообщаться информация о количестве выполненных попыток и о количестве успешных попыток.

3.1.35. Написать программу построения графика функции  $f(x)$  так, чтобы участки графика, на которых  $f(x) > 0$ , и участки, на которых  $f(x) < 0$  окрашивались в разный цвет.

3.1.36. Построить:

- треугольник, положение вершин которого на экране определяется парами чисел 50, 50; 100, 50; и 80, 190;
- четырехугольник, положение вершин которого на экране определяется парами чисел 70, 80; 170, 70; 170, 150 и 80, 140;
- шестиугольник, положение вершин которого на экране определяется парами чисел 120, 100; 140, 120; 140, 140; 120, 160; 100, 140 и 100, 120.

3.1.37. Построить квадрат со стороной 30, центр которого совмещен с центром экрана. Стороны квадрата должны быть параллельны сторонам экрана. (Длины отрезков здесь и далее задаются числом точек).

3.1.38. Построить прямоугольник со сторонами 30 и 50, центр которого совмещен с центром экрана. Стороны прямоугольника должны быть параллельны сторонам экрана.

3.1.39. Дано шесть целых чисел, определяющих положение вершин треугольника, расположенного в левой половине экрана. Построить на экране этот треугольник, а также треугольник, симметричный данному относительно вертикальной прямой, проходящей через середину экрана.

3.1.40. Дано три целых числа, определяющих положение центра окружности на экране и ее радиус. Если окружность не пересекает горизонтальную прямую, проходящую через середину экрана, то вывести данную окружность и окружность, симметричную данной относительно этой прямой.

3.1.41. Четыре целых числа задают положение концов отрезка на экране. Получить изображение этого отрезка и изображение отрезка, центрально-симметричного данному относительно точки, расположенной в центре экрана.

3.1.42. Получить в центре экрана изображение, состоящее из 10 вложенных квадратов со сторонами 10, 20, 30, ..., 100.

3.1.43. Вывести на экран два прямоугольника. Один заштриховать вертикальными прямыми, другой – горизонтальными.

3.1.44. Построить оси координат. Начало координат поместить вблизи левого нижнего угла экрана. Полуоси  $OX$  и  $OY$  направить вправо и вверх соответственно.

3.1.45. Построить 9 концентрических окружностей, окрашенных поочередно в зеленый, красный и коричневый цвета.

3.1.46. Написать программу, в ходе выполнения которой круг зеленого цвета, появившийся в центре экрана и постепенно расширяясь, увеличивается в размерах в три раза, а затем сжимается до начальных размеров.

3.1.47. Пропеллер состоит из двух закрашенных треугольников. Получить на экране врачающийся пропеллер.

3.1.48. Дано два натуральных числа. Написать программу, в ходе выполнения которой отрезок, появившийся в левом верхнем углу экрана, передвинется по экрану так, что его левый конец совместится с точкой, положение которой определяется данными числами. Весь

по струнам гитары вверх и вниз. Мультфильм должен сопровождаться воспроизведением популярной мелодии.

3.1.30. Дано 15 чисел  $a_1, b_1, c_1, a_2, b_2, c_2, \dots, a_i, b_i, c_i$ . Каждая тройка чисел  $a_i, b_i, c_i$  ( $i=1, \dots, 5$ ) определяет параболу  $y = ax^2 + bx + c$ . Построить графики всех этих парабол на отрезке  $[-5, 5]$ . График параболы с номером  $i$  должен иметь цвет с номером  $i \bmod 3 + 1$ .

3.1.31. Написать программу, в ходе выполнения которой одновременно с отысканием приближенного решения уравнения  $f(x)=0$  с помощью алгоритма хорд строится иллюстрация применения этого алгоритма. График функции  $f(x)$ , хорды и вспомогательные вертикальные отрезки должны быть окрашены в разные цвета.

3.1.32. В ходе приближенного вычисления площади криволинейной трапеции получить на экране иллюстрацию применения соответствующего алгоритма.

3.1.33. Метательное устройство находится на расстоянии 800 м от цели. Изменяя начальную скорость полета снаряда  $V_n$  и угол  $\alpha$ , под которым он выпускается, можно регулировать дальность полета. Написать программу с исходными данными  $V_n$  и  $\alpha$ . В левом нижнем углу экрана должен появляться прямоугольник, изображающий метательный аппарат, в правом нижнем углу – отрезок, изображающий цель. Точка, изображающая снаряд, должна переместиться по экрану от аппарата в сторону цели по параболе определяемой исходными данными. В случае попадания снаряда в цель, отрезок, изображающий цель, исчезнет.

3.1.34. Написать игровую программу, позволяющую выполнять несколько попыток при «стрельбе» по цели. В случае попадания в цель она должна менять свое положение на экране. После окончания «стрельбы» должна сообщаться информация о количестве выполненных попыток и о количестве успешных попыток.

3.1.35. Написать программу построения графика функции  $f(x)$  так, чтобы участки графика, на которых  $f(x) > 0$ , и участки, на которых  $f(x) < 0$  окрашивались в разный цвет.

3.1.36. Построить:

- треугольник, положение вершин которого на экране определяется парами чисел 50, 50; 100, 50; и 80, 190;
- четырехугольник, положение вершин которого на экране определяется парами чисел 70, 80; 170, 70; 170, 150 и 80, 140;
- шестиугольник, положение вершин которого на экране определяется парами чисел 120, 100; 140, 120; 140, 140; 120, 160; 100, 140 и 100, 120.

3.1.37. Построить квадрат со стороной 30, центр которого совмещен с центром экрана. Стороны квадрата должны быть параллельны сторонам экрана. (Длины отрезков здесь и далее задаются числом точек).

3.1.38. Построить прямоугольник со сторонами 30 и 50, центр которого совмещен с центром экрана. Стороны прямоугольника должны быть параллельны сторонам экрана.

3.1.39. Дано шесть целых чисел, определяющих положение вершин треугольника, расположенного в левой половине экрана. Построить на экране этот треугольник, а также треугольник, симметричный данному относительно вертикальной прямой, проходящей через середину экрана.

3.1.40. Дано три целых числа, определяющих положение центра окружности на экране и ее радиус. Если окружность не пересекает горизонтальную прямую, проходящую через середину экрана, то вывести данную окружность и окружность, симметричную данной относительно этой прямой.

3.1.41. Четыре целых числа задают положение концов отрезка на экране. Получить изображение этого отрезка и изображение отрезка, центрально-симметричного данному относительно точки, расположенной в центре экрана.

3.1.42. Получить в центре экрана изображение, состоящее из 10 вложенных квадратов со сторонами 10, 20, 30, ..., 100.

3.1.43. Вывести на экран два прямоугольника. Один заштриховать вертикальными прямыми, другой – горизонтальными.

3.1.44. Построить оси координат. Начало координат поместить вблизи левого нижнего угла экрана. Полуоси  $OX$  и  $OY$  направить вправо и вверх соответственно.

3.1.45. Построить 9 концентрических окружностей, окрашенных поочередно в зеленый, красный и коричневый цвета.

3.1.46. Написать программу, в ходе выполнения которой круг зеленого цвета, появившийся в центре экрана и постепенно расширяясь, увеличивается в размерах в три раза, а затем сжимается до начальных размеров.

3.1.47. Пропеллер состоит из двух закрашенных треугольников. Получить на экране врачающийся пропеллер.

3.1.48. Дано два натуральных числа. Написать программу, в ходе выполнения которой отрезок, появившийся в левом верхнем углу экрана, передвинется по экрану так, что его левый конец совместится с точкой, положение которой определяется данными числами. Весь

путь отрезка должен состоять из двух участков – горизонтального и вертикального.

3.1.49. Написать программу, в ходе выполнения которой зеленый квадрат, появившийся в левом верхнем углу экрана, перемещается вправо вниз по диагонали.

3.1.50. Данна последовательность, состоящая из 70 натуральных чисел. Первые тридцать чисел определяют положение на экране 10 окружностей. Остальные 40 чисел – положение на экране 20 точек. Высветить:

- а) все окружности и те из данных точек, которые лежат внутри не более чем одной окружности;
- б) только те окружности, внутри которых содержится хотя бы одна из данных точек;
- в) все точки, не попадающие внутрь окружностей и все окружности, не содержащие ни одной из данных точек.

3.1.51. Даны натуральные  $x_i, y_i$ . Построить на экране точки, задаваемые парами значений  $x_i, y_i$  ( $i=1, \dots, 20$ ) и соединить пары точек, наиболее удаленные друг от друга.

3.1.52. Даны натуральные  $x_i, y_i$ . Построить на экране точки, задаваемые парами значений  $x_i, y_i$  ( $i=1, \dots, 20$ ) и соединить отрезками прямых:

- а) каждую из точек со всеми остальными;
- б) точки с номерами одной четности;
- в) точки с номерами разной четности.

3.1.53. Даны натуральные  $x_i, y_i$ . Построить на экране точки, задаваемые парами значений  $x_i, y_i$  ( $i=1, \dots, 40$ ) и прямоугольник со сторонами, параллельными сторонам экрана, содержащий все эти точки.

## 4.1. ЛИТЕРНЫЕ КОНСТАНТЫ

Литера, заключенная в одиночные кавычки, представляет целое значение, равное коду этой литеры (в кодировке, принятой на данной машине). Это так называемая литерная константа. Существует и другой способ для написания маленьких целых значений. Например, 'A' есть литерная константа; в наборе литер ASCII ее значение равняется 65 — внутреннему представлению буквы A. Конечно, 'A' в роли константы предпочтительнее, чем 65, поскольку смысл первой записи более очевиден, и она не зависит от конкретного способа кодировки литер.

Эскейп-последовательности, используемые в строковых константах, допускаются также и в литерных константах. Так, '\n' обозначает код литеры «новая-строка», который в ASCII равен 10. Следует обратить особое внимание на то, что '\n' обозначает одну литеру (код которой в выражении рассматривается как целое значение), в то время как "\n" — строковая константа, в которой чисто случайно указана одна литерра.

4.1.1. Напишите программу для подсчета пробелов, табуляций и новых-строк.

4.1.2. Напишите программу, копирующую литеры ввода в выходной поток и заменяющую подряд стоящие пробелы на один пробел.

4.1.3. Напишите программу, копирующую вводимые литеры в выходной поток с заменой литеры табуляции на \t, литеры забоя на \b и каждой обратной наклонной черты на \\. Это сделает видимыми все литеры табуляции и забоя.

Программа подсчитывает строки, слова и литеры, причем под словом здесь имеется в виду любой string литер, не содержащий в себе пробелов, табуляций и новых-строк.

```
#include<stdio.h>
#define IN 1 /*внутри слова*/
```

4.3.7. Напишите программу, печатающую литеры входного потока так, чтобы строки текста не выходили правее  $n$ -й позиции. Это значит, что каждая строка, длина которой превышает  $n$ , должна печататься с переносом на следующие строки. Место переноса следует «искать» после последней непробельной литеры, расположенной левее  $n$ -й позиции. Позаботьтесь о том, чтобы ваша программа вела себя разумно в случае очень длинных строк, а также, когда до  $n$ -й позиции не встречается ни одной литеры пробела или табуляции.

4.3.8. Напишите программу, убирающую все комментарии из любой Си-программы. Не забудьте должным образом обработать строки литер и стринговые константы. Комментарии в Си не могут быть вложены друг в друга.

4.3.9. Напишите программу, проверяющую Си-программы на элементарные синтаксические ошибки типа несбалансированности скобок всех видов. Не забудьте о кавычках (одиночных и двойных), эскейп-последовательностях (...) и комментариях. (Это - сложная программа, если ее писать для общего случая.)

4.3.10. Напишите версию функции `squeese(s1,s2)`, которая удаляет из  $s1$  все литеры, встречающиеся в *стринге*  $s2$ .

4.3.11. Напишите функцию `apu(s1,s2)`, которая возвращает либо позицию в  $s1$ , где стоит первая литер, совпадшая с любой из литер в  $s2$ , либо -1 (если ни одна литер  $s1$  не совпадает с литерами из  $s2$ ). (Стандартная библиотечная функция `strpbrk` делает то же самое, но выдает указатель на литеру, а не номер ее позиции.)

## 4.4. ПОБИТОВЫЕ ОПЕРАТОРЫ

4.4.1. Напишите:

а) функцию `setbits(x, p, n, y)`, возвращающую значение  $x$ , в котором  $n$  бит, начиная с  $p$ -й позиции, заменены на  $n$  правых разрядов из  $y$  (остальные биты не изменяются).

б) функцию `setbits(x, p, n, y)`, возвращающую значение  $x$ , в котором  $n$  бит, начиная с  $p$ -й позиции, заменены на  $n$  левых разрядов из  $y$  (остальные биты не изменяются).

4.4.2. Напишите функцию `invert(x, p, n)`, возвращающую значение  $x$  с инвертированными  $n$  битами, начиная с позиции  $p$  (остальные биты не изменяются).

4.4.3. Напишите:

а) функцию `rightrot(x, n)`, которая циклически сдвигает («вращает») вправо  $x$  на  $n$  разрядов.

б) Напишите функцию `lefttrot(x, n)`, которая циклически сдвигает («вращает») влево  $x$  на  $n$  разрядов.

4.4.4. Применительно к числам, в представлении которых использован дополнительный код, выражение  $x \&=(x-1)$  уничтожает самую правую 1 в  $x$ . Объясните, почему. Используйте это наблюдение при написании более быстрого варианта функции `bitcount`.

## 4.5. ПЕРЕКЛЮЧАТЕЛЬ switch

4.5.1. Напишите функцию `escape(s, t)`, которая при копировании текста из  $t$  в  $s$  преобразует литеры типа новая-строка и табуляция в «видимые последовательности литер» (типа `\n` и `\t`). Используйте инструкцию `switch`. Напишите функцию, выполняющую обратное преобразование эскейп-последовательностей в настоящие литеры.

## 4.6. ЦИКЛЫ While, For и Do While

4.6.1. Напишите функцию `expand(s1,s2)`, восстанавливающую сокращенную запись типа `a-z` в стринге  $s1$  до эквивалентной полной записи `abc...xyz` в  $s2$ . В  $s1$  допускаются буквы (прописные и строчные) и цифры. Следует уметь справляться со случаями типа `a-b-c`, `a-0-9` и `-a-b`. Считайте знак — в начале или в конце  $s1$  обычной литературой минус.

Опыт показывает, что цикл `do-while` гораздо реже используется, чем `while` и `for`. Тем не менее, потребность в нем время от времени возникает, как, например, в функции `itoa` (обратной по отношению к `atoi`), преобразующей число в стринг литер. Выполнить такое преобразование оказалось несколько более сложным делом, чем ожидалось, поскольку простые алгоритмы генерируют цифры в обратном порядке. Предлагается остановиться на варианте, в котором сначала формируется цепочка цифр, а затем она реверсируется.

```
/*itoa: преобразование n в стринг s*/
void itoa ( int n, char s[])
{
    int i, sign;
    if ((sign=n)<0) /*сохраняем знак*/
        n=-n; /*делаем n положительным*/
```

```

i=0;
    do { /*генерируем цифры в обратном порядке*/
        s[i++] = n % 10+'0'; /*следующая цифра*/
    } while ((n /= 10)>0); /*исключить ее*/
    if (sign<0)
        s[i++]='-';
    s[i]='\0';
    reverse(s);
}

```

Конструкция *do-while* здесь необходима, или по крайней мере удобна, поскольку в *s* посыпается хотя бы одна литерал, даже если *n* равно нулю. В теле цикла одну инструкцию мы выделили фигурными скобками (хотя они и избыточны), чтобы не принять по ошибке слово *while* за начало цикла *while*.

4.6.2. При условии, что для представления чисел используется дополнительный код, наша версия *itoa* не справляется с самым большим по модулю отрицательным числом, значение которого равняется  $-2^{n-1}$  (где *n* – размер слова). Объясните, чем это вызвано. Модифицируйте программу таким образом, чтобы она давала правильное значение указанного числа независимо от машины, на которой выполняется.

4.6.3. Напишите функцию *itob(n, s, b)*, которая переводит целое *n* в строку *s*, представляющий число по основанию *b*. В частности, *itob(n, s, 16)* должна помещать в *s* текст числа *n* в шестнадцатиричном виде.

4.6.4. Напишите версию *itoa* с дополнительным третьим аргументом, задающим минимальную ширину поля. При необходимости преобразованное число слева должно дополняться пробелами.

4.6.5. Напишите функцию *strrindex(s, t)*, которая выдает позицию самого правого вхождения *t* в *s* или *-1*, если вхождения не обнаружено.

4.6.6. Напишите функцию *atof(s)*, которая переводит строку *s* в соответствующее число с плавающей точкой двойной точности таким образом, чтобы она справлялась с числами вида *123.45e-6* в которых в конце может стоять *e* (или *E*) с последующей экспонентой (быть может, со знаком).

*Напишите программу-калькулятор с обратнойпольской записью*

```

#include <stdio.h>
#include <stdlib.h> /*для atof()*/
#define MAXOP 100 /*макс.размер операнда или оператора*/

```

```

#define NUMBER '0' /*признак числа*/
int getop(char []);
void push(double);
double pop(void);
main()
int type;
double op2;
char s[MAXOP];
while ((type = getop(s)) != EOF)
{ switch (type)
(case NUMBER:
    push(atof(s));
    break;
case '+':
    push(pop() + pop());
    break;
case '*':
    push(pop() * pop());
    break;
case '-':
    op2=pop();
    push(pop() - op2);
    break;
case '/':
    op2=pop();
    if (op2 != 0.0)
        push(pop() / op2);
    else
        printf("ошибка: деление на нуль\n");
    break;
case '\n':
    printf("\t%.8g\n", pop());
    break;
default:
    printf("ошибка: неизвестная операция %s\n", s);
    break;
return 0;
}

#define MAXVAL 100 /*максимальная глубина стека*/
int sp = 0; /*позиция след. свобод. эл-та стека*/

```

```

double val[MAXVAL]; /*стек*/
/*push: положить значение f в стек*/
void push(double f)
{
    if (sp < MAXVAL)
        val[sp++]=f;
    else
        printf("ошибка: стек полон, %d не помещается\n", f);
    /*pop: взять с вершины стека и выдать в качестве результата*/
    double pop(void)
    {
        if (sp>0)
            return val[-sp];
        else {
            printf("ошибка: стек пуст\n");
            return 0.0;
        }
    }
}

```

```

#include <ctype.h>
int getch(void);
void ungetch(int);
/*getop: получает следующий оператор или operand*/
int getop(char s[])
{
    int i, c;
    while ((s[0]=c=getch())==' ' || c=='\t')
        s[1]='\0';
    if (!isdigit(c) && c != '.')
        return c; /*не число*/
    i=0;
    if (isdigit(c)) /*накапливаем целую часть*/
        while (isdigit(s[++i]=c=getch()))
            if (c=='.')
                /*накапливаем дробную часть*/
                while (isdigit(s[++i]=c=getch()))
                    s[i]='\0';
    if (c!=EOF) ungetch(c);
    return NUMBER;
}

```

```

#define BUFSIZE 100
char buf[BUFSIZE], /*буфер для ungetc*/
int bufp=0; /*след. свободная позиция в буфере*/
int getch(void) /*дай (возможно возвращенную) литеру*/
{
    return (bufp>0) ? buf[-bufp] : getchar();
}

void ungetch(int c) /*верни литеру на ввод*/
{
    if (bufp >= BUFSIZE)
        printf("ungetch: слишком много литер\n");
    else
        buf[bufp++]=c;
}

```

4.6.7. Исходя из предложенной нами схемы, дополните программу-калькулятор таким образом, чтобы она «понимала» оператор взятия модуля (%) и отрицательные числа.

4.6.8. Добавьте команды, с помощью которых можно было бы печатать верхний элемент стека (с сохранением его на стеке), дублировать его на стеке, менять местами два верхних элемента стека. Введите команду очистки стека.

4.6.9. Предусмотрите возможность использования в программе библиотечных функций sin, exp и pow.

4.6.10. Введите команды для работы с переменными (легко обеспечить до 26 переменных, каждая из которых имеет имя, представленное одной буквой латинского алфавита). Добавьте переменную, предназначенную для хранения самого последнего из напечатанных значений.

4.6.11. Напишите программу ungets(s), возвращающую строку s во входной поток. Должна ли ungets «знать» что-либо о переменных buf и bufp или ей достаточно пользоваться только функцией ungetch?

4.6.12. Предположим, что число литер, возвращаемых назад, не превышает 1. Модифицируйте с учетом этого факта функции getch и ungetch.

4.6.13. В функциях не предусмотрена возможность возврата EOF. Подумайте, что надо сделать, чтобы можно было возвращать EOF, и скорректируйте соответственно программу.

4.6.14. В основу программы калькулятора можно положить применение функций getline, которая читает целиком строку; при этом отпадает необходимость в getch и ungetch. Напишите программу, реализующую этот подход.

## 4.7. РЕКУРСИЯ

4.7.1. Преобразуйте целое число в цепочку цифр с помощью рекурсивной программы.

4.7.2. Напишите рекурсивную версию функции `reverse(s)`, обрабатывающую строку на его же месте.

## 4.8. МАКРОСЫ

4.8.1. Определите в виде макроса:

- а) `swap(i, x, y)`, который осуществляет обмен значениями указанного типа *i* между аргументами *x* и *y*;
- б) `max(x, y)`;
- в) `max(x, y, z)`;
- г) `min(x, y)`;
- в) `min(x, y, z)`;
- е) `abs(x)`;
- ж) `sign(x)`, который определяет знак числа и возвращает 1, 0, -1.

(Примените блочную структуру.)

## 4.9. УКАЗАТЕЛИ И МАССИВЫ

Напишите программу перестановки `*px` и `*py`

```
void swap(int *px, int *py)
{
    int temp;
    temp=*px;
    *px=*py;
    *py=temp;
}
```

Напишите функцию `getint`, которая осуществляет ввод в свободном формате одного целого числа и его перевод из текстового представления в значение типа `int`. Функция `getint` должна возвращать значение полученного числа или сигнализировать значением `EOF` о конце файла, если входной поток исчерпан. Эти значения должны возвращаться по разным каналам, так как нельзя рассчитывать на то, что полученное в результате перевода число никогда не совпадет с `EOF`.

```
#include <ctype.h>
int getch(void);
void ungetch(int);
/*getint: читает следующее целое из ввода в *pn*/
int getint(int *pn)
{
    int c, sign;
    while (isspace(c=getch()));
    /*пропуск пробельных литер*/
    if (!isdigit(c)&& c!=EOF&&c!='+'&&c!='-') {
        ungetch(c); /*не число*/
        return 0;
    }
    sign=(c=='-') ? -1 : 1;
    if (c=='+' || c=='-')
        c=getch();
    for (*pn=0; isdigit(c); c=getch())
        *pn=10 * *pn+(c-'0');
    *pn *=sign;
    if (c!=EOF)
        ungetch(c);
    return c;
}
```

4.9.1. Функция `getint` написана так, что знаки – или +, за которыми не следует цифра, она понимает как «правильное» представление нуля. Скорректируйте программу таким образом, чтобы она в подобных случаях возвращала прочитанный знак назад во ввод.

4.9.2. Напишите функцию `getfloat` – аналог `getint` для чисел с плавающей точкой. Какой тип будет иметь результирующее значение, выдаваемое функцией `getfloat`?

Напишите функцию `strlen`, которая возвращает длину строки

```
int strlen (char *s)
{
    int n;
    for (n=0; *s]!='\0'; s++)
        n++;
    return n;
}
```

4.9.3. Используя указатели, напишите функцию `strcat(s, t)`, которая копирует строку `t` в конец строки `s`.

4.9.4. Напишите функцию `strend(s, t)`, которая выдает 1, если строка `t` расположен в конце строки `s`, и нуль в противном случае.

4.9.5. Напишите варианты библиотечных функций `strncpy`, `strncat` и `strncmp`, которые оперируют с первыми литерами своих аргументов, число которых не превышает `n`. Например, `strncpy(t, s, n)` копирует не более `n` литер `t` в `s`.

## 4.10. АРГУМЕНТЫ КОМАНДНОЙ СТРОКИ

Напишите программу поиска образца, образец для поиска задаётся первым аргументом в командной строке

```
#include<stdio.h>
#include<string.h>
#define MAXLINE 1000
int getline(char *line, int max);
main(int argc,char *argv[])
{ char line[MAXLINE];
  int found = 0;
  if (argc != 2)
    printf("Используйте в find образец\n");
  else
    while (getline(line, MAXLINE)>0)
      if (strstr(line, argv[1]) != NULL){ printf ("%s",
line);
        found++;
      }
  return found;
}
/*find: печать строк по образцу из 1-го аргумента*/
main(int argc, char *argv[])
{ char line[MAXLINE];
  long lineno = 0;
  int c, except = 0, number = 0, found = 0;
  while (-argc>0 && (*++argv[0] == '-')
```

```
while (c=++argv[0])
switch (c) {
case 'x':
except = 1;
break;
case 'n':
number = 1;
break;
default:
printf("find: неверный параметр. %c\n", c);
argc = 0;
found = -1;
break;
if (argc != 1)
printf("Используйте: find -x-n образец\n");
else
while (getline(line,MAXLINE)>0)
{ lineno++;
  if ((strstr(line, *argv)!=NULL) != except) {
    if (number)
      printf("%ld: ", lineno);
    printf("%s",line);
    found++;
  }
}
return found;
}
```

4.10.1. Напишите программу `expr`, интерпретирующую обратную польскую запись выражения, задаваемую командной строкой, в которой каждый оператор и operand представлены отдельным аргументом. Например,

`expr 2 3 4 + *` вычисляется так же, как выражение `2x(3+4)`.

4.10.2. Усовершенствуйте программы `entab` и `dentab` таким образом, чтобы через аргументы можно было задавать список «стопов» табуляции.

4.10.3. Расширьте возможности таким образом, чтобы при обращении вида

`entab -m +n`

«стопы» табуляции начинались с  $m$ -й позиции и выполнялись через каждые  $n$  позиций. Разработайте удобный для пользователя вариант поведения программы по умолчанию (когда нет никаких аргументов).

4.10.4. Напишите программу `tail`, печатающую  $n$  последних введенных строк. По умолчанию значение  $n$  равно 10, но при желании  $n$  можно задать с помощью аргумента. Обращение вида

```
tail -n
```

печатает  $n$  последних строк. Программа должна вести себя осмысленно при любых входных данных и любом значении  $n$ . Напишите программу так, чтобы наилучшим образом использовать память; запоминание строк организуйте, используя динамическое расширение памяти.

4.10.5. Напишите программу сортировки, чтобы она реагировала на параметр `-g`, указывающий, что объекты нужно сортировать в обратном порядке, т.е. в порядке убывания. Обеспечьте, чтобы `-g` работал и вместе с `-n`.

4.10.6. Введите в программу необязательный параметр `-f`, задание которого делало бы неразличимыми литеры нижнего и верхнего регистров (например, *a* и *A* должны оказаться при сравнении равными).

4.10.7. Предусмотрите в программе необязательный параметр `-d`, который заставит программу при сравнении учитывать только буквы, цифры и пробелы. Организуйте программу таким образом, чтобы этот параметр мог работать вместе с параметром `-f`.

4.10.8. Реализуйте в программе возможность работы с полями: возможность сортировки по полям внутри строк. Для каждого поля предусмотрите свой набор параметров.

4.10.9. Напишите программу, которая читает текст Си-программы и печатает в алфавитном порядке все группы имен переменных, в которых совпадают первые 6 литер, но последующие в чем-то различаются. Не обрабатывайте внутренности строков и комментариев. Число 6 сделайте параметром, задаваемым в командной строке.

4.10.10. Напишите программу печати таблицы «перекрестных ссылок», которая будет печатать все слова документа и указывать для каждого из них, номера строк, где оно встретилось. Программа должна игнорировать «шумовые» слова типа «и», «или» и т.д.

4.10.11. Напишите программу, которая печатает весь набор различных слов, образующих входной поток, в порядке возрастания час-

88

тоты их встречаемости. Перед каждым словом должно быть указано число вхождений.

4.10.12. Напишите программу, которая будет печатать разумным способом любой ввод. Как минимум она должна уметь печатать неграфические литеры в восьмеричном или шестнадцатиричном виде (в форме, принятой на вашей машине), обрывая длинные текстовые строки.

4.10.13. Напишите основанную на постфиксной записи программу калькулятора таким образом, чтобы для ввода и преобразования чисел она использовала `scanf` и/или `sscanf`.

## ГЛАВА 5

### ЗАДАЧИ ПО ОБЪЕКТНО-ОРИЕНТИРОВАННОМУ ПРОГРАММИРОВАНИЮ

Объектно-ориентированное программирование является еще одним методом декомпозиции сложных задач. Этот метод заключается в представлении задачи в виде функционирования совокупности взаимодействующих объектов. Такое представление задачи является наиболее естественным для человека, однако, для каждой конкретной задачи необходимо определить целесообразность того или иного метода ее декомпозиции. Введение в программирование понятия класса объектов позволяет исключить значительное дублирование информации за счет «наследования» подклассами свойств (структур и методов) базовых классов (классов, структура и поведение которых «наследует» другой класс).

Класс представляет собой уровень абстракции – совокупность признаков (как структурных, так и функциональных), позволяющих отделить объекты одного класса от объектов любого другого класса. При этом игнорируются признаки, являющиеся внутриклассовыми различиями, и выделяются признаки, позволяющие отделить один класс объектов от другого (т.е. признаки, позволяющие определить, что данный объект относится к одному, а не другому классу).

Принцип наследования структуры суперкласса позволяет избежать описания аналогичных структур в подклассах, поскольку каждый подкласс (объявленный как производный от некоторого суперкласса) неявно содержит описание структуры суперкласса.

Поскольку возможности языка Си++ значительно расширены по отношению к языку Object Pascal, принципы объектно-ориентированного программирования будут рассматриваться на основе языка Си++.

#### 5.1. ВИРТУАЛЬНЫЕ ФУНКЦИИ

Как правило, многие классы, наследующие структуру и методы своего суперкласса, не только дополняют, но и содержат несколько измененные методы суперкласса. Возможность переопределения мето-

дов базового класса в производных классах обеспечивается механизмом виртуальных функций.

Виртуальная функция – это функция, объявленная с ключевым словом `virtual` в базовом классе и переопределенная в производных классах. В отличие от обычных функций, адрес виртуальных функций вычисляется во время выполнения программы и основывается на типе объекта. Таким образом, для различных объектов могут вызываться различные версии одной и той же виртуальной функции.

Для классов «Точка» и «Окружность» определить виртуальную функцию `Draw` прорисовки объекта, функцию элементарного перемещения с заданной скоростью в заданном направлении.

```
#include<graphics.h>
class Point
{
    int x,y; // координаты
    int color; // цвет
    float v; // скорость
    float alpha; // направление движения
public:
    Point(int _x,int _y,int _color,float _v=10,
          float_alpha=M_PI/3):
        x(_x),y(_y),color(_color),
        v(_v),alpha(_alpha){Draw(color);}
    virtual void Draw(int col);
    void El_Move();
    void Run();
};
void Point::Draw(int col)
{
    putpixel(x,y,col);
}
void Point::El_Move()
// функция осуществляет элементарное изменение координат со
// скоростью v
{
    x+=ceil(v*cos(alpha));
    y+=ceil(v*sin(alpha));
}
void Point::Run()
```

```

do
{
    Draw(0);
    El_Move();
    Draw(color);
}while(!kbhit());
}
class Circ: public Point
{
    int rad;
public Circ((int _x,int _y,int _color,float _v,
float _alpha,int _rad):
    Point(_x,_y,_color,_v,_alpha),rad(_rad){
Draw(_color);}
void Draw(int col);
};
void Circ::Draw(int col)
{
    setcolor(col);
    circle(x,y,rad);
}
int main()
{
    int gd=DETECT,gm;
    initgraph(&gd,&gm,"c:\\borlandc\\bgi");
    Point point (20,20,1);
    Circ circ(200,200, 4,15,M_PI/6);
    point.Run();
    circ.Run();
    closegraph();
    return 0;
}

```

Для объектов *point* и *circ* вызывается одна и та же функция базового класса *Run()*, однако, в ней для первого объекта вызывается функция *Draw(int)* базового класса, для второго – функция *Draw(int)* производного класса *Circ*.

5.1.1. Создать иерархическую структуру классов «точка», «четырехугольник», «линия», «окружность», «полигон». Определить для них следующие операции:

92

- прорисовки;
- перемещения с заданной скоростью в заданном направлении;
- вращения вокруг заданной точки;
- масштабирования.

Организовать движение этих фигур до нажатия произвольной клавиши.

5.1.2. Разработать программу, моделирующую поведение двух стай рыб, одна из которых является хищником для другой. Представители обеих стай могут «видеть» друг друга в заданном радиусе, изменять в определенных пределах скорость и направление движения. Для стай определены коэффициенты естественной смертности и рождаемости. «Хищники» могут преследовать и «поедать» «жертв», если окажутся в непосредственной близости. Для визуализации модели воспользоваться классами, определенными в задаче 5.1.1.

5.1.3. Создать иерархию классов для моделирования взаимодействия «Магазин-склад-Заказчик». В распоряжении магазина имеется:

- склад с товарами различных наименований, стоимости и веса;
- гараж с машинами различной грузоподъемности, вместительности и потребляемой мощности.

В магазин поступают заказы на поставку определенного количества товаров различных наименований в заданный район (задается как расстояние между магазином и районом). Магазин должен выбрать наиболее подходящее транспортное средство (учитывать, что машины могут быть заняты другими заказчиками) и определить счет за поставку товара.

5.1.4. Создать иерархию классов следующих элементов интерфейса: «Окно», «Кнопка», «Меню», «Окно редактирования», «Окно вывода текста или графики», «Кнопка-переключатель», «Кнопка-активизатор», «Радиокнопка». С помощью созданных классов построить интерфейс к задаче 5.1.2.

5.1.5. Создать иерархию классов и запрограммировать компьютерную игру «Морской бой». Игровое поле разбивается на клетки. Палуба корабля занимает одну клетку. Кораблем считается совокупность смежных клеток. Допускается четыре категории кораблей: одно-, двух-, трех- и четырехпалубные. Двух-, трех- и четырехпалубные корабли могут иметь всевозможные конфигурации и ориентации на игровом поле. Не допускается размещение кораблей в смежных клетках.

## 5.2. ПЕРЕГРУЗКА ОПЕРАЦИЙ

Механизм Си++, называемый перегрузкой операций, позволяет манипулировать классами как встроенными типами данных, т.е. переопределять для новых классов операции, определенные для встроенных типов (такие как «+», «-» и т.д.).

Для перегрузки операций используется следующий синтаксис:

```
<тип возвращаемого значения> operator <оператор> (<список
параметров-операндов>)
{
    <тело функции-оператора>
}
```

В случае, если описание оператора выносится за пределы класса, перед словом «operator» аналогично методам класса указывается принадлежность классу.

Реализовать перегрузку операций «+=» и «<<» для класса «Очередь целых чисел».

```
struct el // тип элементов списка
{
    int i;
    el *next;
};

class spis
{
    el * head; // указатель на начало списка
    el * end; // указатель на конец списка
    int n; // количество элементов списка
public:
    spis(){head=0;end=0;} // конструктор класса
    ~spis(); // деструктор
    spis & operator +=(spis a); // прототип функции пере-
    грузки операции «+=» для // добавления списка
    spis & operator +=(int a); // прототип функции пере-
    грузки операции «+=» для // добавления элемента
};

spis::~spis()
{
    while(head)
    {
```

```
        el * cur=head;
        head=head->next;
        delete cur;
    }
}

spis & spis::operator +=(spis a)
{
    end->next=a.head;
    while(end->next) end=end->next;
    return *this;
}

spis & spis::operator +=(int a)
{
    if(head)
    {
        end->next=new el;
        end=end->next;
    }
    else
    {
        head=new el;
        end=head;
    }
    end->i=a;
    end->next=0;
    return *this;
}
```

5.2.1. Создать класс «Список\_целых\_чисел» и переопределить для него операции:

- а) «[ ]» – взятие элемента списка с заданным номером;
- б) «->» – удаление элемента с заданным значением;
- в) «==» – сравнение на равенство двух списков;
- г) «<<», «>>» – сравнение длин списков;
- д) «\*» – слияние двух упорядоченных списков в один, не нарушая упорядоченности.

5.2.2. Создать класс «Стек\_целых\_чисел», производный от класса «Список\_целых\_чисел», переопределить для него операции:

- а) «+=» – добавление к стеку элемента или другого списка;
- б) «-->» – удаление верхнего элемента стека;
- в) «>>» – ввод элементов стека с консоли;

г) «<<>» – вывод элементов стека на экран или в файл.

5.2.3. Переопределить операции, указанные в задаче 5.2.1, для класса «Очередь\_целых\_чисел», производного от класса «Список\_целых\_чисел».

5.2.4. Создать класс «Вектор\_целых\_чисел», переопределить для него операции:

- а) «+=» – сложение векторов с сохранением значения в одном из них;
- б) «+» – сложение векторов с сохранением значения в новом векторе;

в) «-» – разность векторов;

г) «[]» – доступ к заданному элементу;

д) «>>» – ввод элементов вектора с консоли;

е) «<<>» – вывод элементов вектора на экран или в файл.

5.2.5. Создать класс «Матрица\_целых\_чисел», используя класс «Вектор\_целых\_чисел» с переопределенными для него операциями. Определить для этого класса следующие операции:

- а) «>>» – ввод матрицы с консоли;
- б) «<<>» – вывод матрицы на экран или в файл;
- в) «+» – сложение матриц;
- г) «-» – вычитание матриц;
- д) «\*» – умножение матриц;
- е) «[ ][ ]» – доступ к элементу;
- ж) функцию `det` вычисления определителя матрицы;
- з) функцию `ort` проверки ортогональности строк матрицы;
- и) функцию `transp` получения транспонированной матрицы;
- к) функцию `revers` получения обратной матрицы;
- л) функцию `self_value` получения собственных значений матрицы;
- м) функцию `self_vector` получения собственных векторов матрицы.

5.2.6. Создать класс «Матрица\_целых\_чисел» и определить для него операции, указанные в задаче 5.2.5, не используя класс «Вектор\_целых\_чисел».

5.2.7. Используя класс «Матрица\_целых\_чисел» и определенные для него операции, создать класс «Система линейных уравнений» и определить для него операции:

- а) решение системы методом Гаусса;
- б) решение системы методом Крамера.

5.2.8. Создать класс «Функция», определить для него операции:

а) «>>>» – ввод функции в виде текстовой строки с консоли;

б) «<<>» – вывод функции в виде текстовой строки на экран;

в) «( )» – вычисление значения функции в точке по дереву-формуле;

г) «++» – получение дерева-функции по функции, заданной в виде текстовой строки;

д) «- -» – получение представления функции в виде текстовой строки по ее дереву-формуле;

е) функцию `dif` получения дерева-производной дерева-функции;

ж) функцию `graphic`, выводящую график функции на заданном интервале в заданном окне.

5.2.9. Создать класс «Строка\_Си», определить для него операции:

а) «+» – слияния строк;

б) «==» – сравнения строк на равенство;

в) «=» – копирования строки;

г) «.-» – удаления из строки подстроки;

д) «/» – удаления из строки всех символов, содержащихся в строке-делителе;

е) функцию `at(char * substr, char * dest, int i)`, возвращающую номер позиции  $i$ -того вхождения строки `substr` в строку `dest`.

### 5.3. ШАБЛОНЫ

В задачах конструирования классов «Матрица», «Вектор», «Список» и др. необходимо было определить тип элементов этих структур. Однако большинство операций, определенных для этих структур не зависит от типа их элементов. Тем не менее, для этих же структур с другими типами элементов необходимо было бы определить собственные классы с аналогичными операциями, такие, например, как «Список\_чисел\_с\_плавающей\_точкой». Во избежание таких ситуаций в языке Си введено понятие шаблона.

Шаблон — это обобщенное описание семейства классов, имеющих общую структуру и операции, но использующих различные типы и константы.

На основе описания шаблона можно генерировать новый класс для конкретного типа или константы.

Шаблон определяется следующим синтаксисом:

`template <список аргументов>`

```
class <имя>
{
    <тело класса>
};
```

где список аргументов может содержать определение произвольного типа или константы.

Для определения произвольного типа используется следующий синтаксис:

```
class <идентификатор> (например, class T);
для определения констант –
<тип><идентификатор> (например, int size).
```

Элементы списка отделяются друг от друга запятой.

В случае описания методов шаблона за его пределами необходимо перед описанием каждого метода включать заголовок шаблона *template <список аргументов>*. Заголовок шаблона также необходимо включать перед любым употреблением имени класса шаблона.

Создать шаблон класса «Список».

```
template <class el>
struct spis
{
    el * head;
    el * end;
    int n;
    spis():head=0,end=0;
    friend ostream & operator <<(ostream&out,spis &a);
    el & operator [](int i)
    {};
    template <class el>
    el & spis::operator [](int i)
    {
        el * cur=head;
        for(int j=1;j<i;j++)
            cur=cur->next;
        return cur->i;
    }
    template <class el>
    ostream & operator <<(ostream & out,spis &a)
    {
        el * cur=a.head;
```

```
        for(cur=a.head; cur; cur=cur->next)
            out <<cur->i <<" ";
        return out;
    }
```

Создание объекта класса «списка элементов с плавающей точкой» производится следующим образом: *spis<float> spisok\_of\_floats;* список целых чисел — *spis<int> spisok\_of\_ints*.

5.3.1. Построить шаблоны классов:

- а) «Матрица\_произвольных\_элементов»;
- б) «Вектор\_произвольных\_элементов»;
- в) «Очередь\_произвольных\_элементов»;
- г) «Стек\_произвольных\_элементов».

5.3.2. Используя шаблон «Стая», решить задачу 5.1.1 (моделирование взаимодействия двух стай рыб).

## ЛИТЕРАТУРА

1. Абрамов С.А., Зима Е.В. Начала программирования на языке Паскаль. – М.: Наука, 1989.
2. Ахо А., Хонкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. – М.: Мир, 1979.
3. Баррон Д. Рекурсивные методы в программировании. – М.: Мир, 1974.
4. Бородич Ю.С., Вальвачёв А.Н., Кузьмич А.И. Паскаль для персональных компьютеров. Справ. Поссобие. – Мин.: Выш. Шк.: БФ ГИТМП «НИКА», 1991.
5. Вирт Н. Алгоритмы и структуры данных. – М.: Мир, 1989.
6. Гудман С., Хидетицели С. Введение в разработку и анализ алгоритмов. – М.: Мир, 1981.
7. Дейтэл Х.М., Дейтэл П. Дж. Как программировать на С++. Пер. с англ. – М.: ЗАО «Издательство БИНОМ», 1998.
8. Катленд Н. Вычислимость. Введение в теорию рекурсивных функций. – М.: Мир, 1983.
9. Кнут Д. Искусство программирования для ЭВМ. Т. 3. – М.: Мир, 1981.
10. Минакова Н.И., Невская Е.С., Угольницкий Г.А., Чекулаева А.А., Чердынцева М.И. Методы программирования. – М.: Вузовская книга, 1999.
11. Пильщиков В.Н. Сборник упражнений по языку Паскаль. – М.: Наука, 1989.
12. Прайс Д. Программирование на языке Паскаль: Практическое руководство. Пер. с англ. – М.: Мир, 1987.
13. Семашко Г.Л., Салтыков А.И. Программирование на языке Паскаль. – М.: Наука, гл. ред. физ.-мат. лит., 1988.

## СОДЕРЖАНИЕ

ГЛАВА 1. Основные приемы программирования . . . . .	3
1.1. Простейшие программы. Оператор присваивания. Операторы ввода-вывода. Простейшая целочисленная арифметика . . . . .	3
1.2. Арифметика вещественных чисел. Вычисления по формулам . . . . .	4
1.3. Логический тип. Разветвления. Условный оператор. Селективный оператор. Составной оператор . . . . .	5
1.4. Операторы цикла. Вложенные операторы цикла . . . . .	8
1.5. Вычисления с заданной точностью . . . . .	12
1.6. Символьный тип. Обработка последовательностей символов . . . . .	13
1.7. Процедуры. Функции . . . . .	14
1.8. Параметры-процедуры и параметры-функции . . . . .	16
ГЛАВА 2. Структуры данных и алгоритмы . . . . .	19
2.1. Массив как структура данных. Векторы . . . . .	19
2.2. Матрицы . . . . .	23
2.3. Вычисления с хранением последовательности значений . . . . .	26
2.4. Сортировка массивов . . . . .	27
2.5. Строки . . . . .	29
2.6. Записи . . . . .	31
2.7. Множества . . . . .	34
2.8. Списки . . . . .	36
2.9. Стеки . . . . .	42
2.10. Очереди . . . . .	46
2.11. Двусвязные списки . . . . .	48
2.12. Деревья . . . . .	52
2.13. Типизированные файлы . . . . .	55
2.14. Текстовые файлы . . . . .	59
2.15. Нетипизированные файлы . . . . .	61
2.16. Сортировка файлов . . . . .	62
2.17. Рекурсия . . . . .	66
2.18. Использование стандартных и собственных модулей при создании программных комплексов . . . . .	68
ГЛАВА 3. Графика и анимация . . . . .	69
ГЛАВА 4. Задачи, демонстрирующие особенности языка Си . . . . .	75
4.1. Литерные константы . . . . .	75
4.2. Массивы . . . . .	76
4.3. Массивы литер . . . . .	77
4.4. Побитовые операторы . . . . .	78
4.5. Переключатель Switch . . . . .	79
4.6. Циклы While, For и Do While . . . . .	79

4.7. Рекурсия . . . . .	84
4.8. Макросы . . . . .	84
4.9. Указатели и массивы . . . . .	84
4.10. Аргументы командной строки . . . . .	86
<b>ГЛАВА 5. Задачи по объектно-ориентированному программированию . . . . .</b>	<b>90</b>
5.1. Виртуальные функции . . . . .	90
5.2. Перегрузка операций . . . . .	94
5.3. Шаблоны . . . . .	97
<b>ЛИТЕРАТУРА . . . . .</b>	<b>100</b>

АМЕЛИНА Н.И., ДЕМЯНЕНКО Я.М., ЛЕБЕДИНСКАЯ Е.Н.,  
ПАСЕЧНЫЙ Л.Г., СПИВАК И.Г., УСОВ А.Б.

### ЗАДАЧИ ПО ПРОГРАММИРОВАНИЮ

Под редакцией  
доктора физико-математических наук,  
профессора Угольницкого Г.А.

Компьютерная верстка И.И. Шильштейна

---

Лицензия на издательскую деятельность № 071370 от 30.12.1996 г.

Сдано в набор 12.05.2000. Подписано в печать 28.07.2000.

Формат 60×84 1/16. Печ. л. 6,5. Тираж 1000.

---

Издательство «Вузовская книга»  
125871, Москва, Волоколамское шоссе, д. 4  
Тел./факс 158-02-35  
E-mail: vbook@mail.ru

Отпечатано ООО "Связь-Принт" в типографии "Радио и связь",  
103473 Москва, 2-й Щемиловский пер., 4/5

*Издательство «Вузовская книга»  
специализируется на выпуске  
учебной, методической, научной  
и научно-популярной литературы.*

Приглашаем к сотрудничеству  
торговые организации,  
а также авторов и спонсоров.

**Обращаться по адресу:**

125871, Москва, Волоколамское шоссе, д. 4.  
Главный административный корпус МАИ,  
комн. 406

Тел.: 158-02-35

E-mail: vbook@mai.ru