Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ

Кафедра компьютерных систем в управлении и проектировании (КСУП)

Е.А. Потапова

ПРОГРАММИРОВАНИЕ

Учебное методическое пособие

2013

Корректор: Осипова Е.А.

Потапова Е.А.
Программирование: учебное методическое пособие. — Томск: Факультет дистанционного обучения, ТУСУР, 2013. — 88 с.

В пособие внесены изменения в 2020 г.

ОГЛАВЛЕНИЕ

КОНТРОЛЬ ОБУЧЕНИЯ

В рамках изучения дисциплины «Программирование» предполагается выполнение лабораторных работ.

Каждое контрольное задание в составе лабораторных работ состоит из нескольких задач, требующих разработки программ на Паскале (можно использовать PascalABC). Использование Delphi не допускается. Разработанные и отлаженные **программы** (*обязательно сопровождающиеся комментариями в тексте*), а также **отчет** по каждой лабораторной работе студент по мере освоения соответствующих разделов языка программирования отправляет на проверку. По результатам проверки работ студенту отправляется рецензия, в которой преподавателем приводится описание ошибок программ (в случае их наличия).

Выбор варианта лабораторных работ осуществляется по общим правилам с использованием следующей формулы:

$$V = (N*K) \text{ div } 100,$$

где V — искомый номер варианта,

N — общее количество вариантов,

div — целочисленное деление (после деления дробная часть отбрасывается), при V = 0 выбирается максимальный вариант,

K — код варианта.

ЛАБОРАТОРНАЯ РАБОТА № 1

Лабораторная работа № 1 посвящена созданию программ с использованием простых управляющих структур: условного оператора, цикла. В задании используются простые типы данных (**нет необходимости использовать массивы)**. Поэтому программы, написанные с использованием массивов, не засчитываются. Кроме того, программы должны быть разработаны в рамках структурного программирования. В частности, запрещается использовать операторы перехода и метки.

Задание состоит из двух задач. Решение первой задачи предусматривает использование простых типов данных, а решение второй задачи — строкового типа данных. В ходе выполнения лабораторной работы необходимо составить программы на языке Паскаль. В некоторых программах полезно определить вспомогательные функции или процедуры. Во всех задачах строки вводятся пользователем с клавиатуры.

Многие задачи лабораторной работы № 1 имеют вид: «Дана последовательность из n (действительных) целых чисел. Определить (вычислить) …» или «Даны натуральное n и вещественные числа a_1, a_2, ..., a_n. Определить (вычислить) …» и т.п. Во всех этих задачах не требуется хранения исходных последовательностей значений. Вводится *n*, затем в цикле, работающем *n* раз, осуществляется пошаговый ввод чисел, и определенным образом постепенно вычисляется необходимый результат.

Примеры решения задач

Простые типы данных

Задача 1.1. *Вычислить: y* = sin1 + sin1.1 + sin1.2 + ... + sin2.

Анализируя данную формулу, видим, что каждое слагаемое данной суммы можно рассчитать по формуле sin(1 + 0.1 * i), где i изменяется от 0 до 10. Поэтому для решения данной задачи можно составить следующий алгоритм.

Переменные:
i — параметр цикла;
y — сумма.

Алгоритм решения задачи:

1. Обнуляем начальное значение переменной y – строка 6, в которой будем накапливать сумму.

2. Организуем цикл для определения суммы (параметр данного цикла должен измениться от 0 до 10) .

3. В данном цикле определяем очередное слагаемое по формуле и добавляем это слагаемое в сумму (строка 7).

4. Выводим результат на экран (строка 8).

```
1)      var y : real;
2)          i : integer;
3)      Begin
4)          writeln('Полученное значение расчета формулы ',
5)                  'y=sin1+sin1.1+sin1.2+ ... +sin2 = ');
6)          y:=0;
7)          for i:=0 to 10 do y:=y+sin(1+0.1*i);
8)          writeln(y);
9)      end.
```

Задача 1.2. *Вычислить:* $y = 1*3*5* \ldots *(2n-1), n>0;$

```
var y : real;
    i, n : integer;
begin
    writeln('Введите количество чисел');
     readln(n);
     y:=1;
     for i:=1 to n do y:=y*(2*i–1);
     writeln('Полученное значение y= ', y)
end.
```

Задача 1.3. *Дано натуральное число N. Разложить его на простые множители.*

Переменные:
n – исследуемое число;
i, j – переменные циклов;
f – вспомогательный флаг.

Алгоритм решения задачи:

1. Вводим значение переменной n. Так как пользователь может случайно ввести отрицательное число, то необходимо дать ему возможность для повторного ввода значения переменной n. Поэтому организуем цикл (строки 3—6 текста программы). Лучше использовать цикл с постпроверкой условия (Repeat…Until). Тело цикла составляют два оператора: оператор вывода на экран приглашения для ввода значения переменной n (строка 4) и оператор чтения с клавиатуры — для непосредственного ввода значения переменной n (строка 5). Данный цикл будет выполняться до тех пор, пока пользователь не введет любое положительное число (строка 6).

2. Выводим на экран значение переменной n и начинаем формировать ответ. Ответ будет представлен в следующем виде (например, в качестве значения переменной n ввели 8): 8 = 1*2*2*2. Так как единица является

простым множителем для любого числа, то выводим ее на экран (строка 7). В результате выполнения данной строки на экране появится: 8 = 1.

3. Вспомогательной переменной f присваиваем значение false. Данная переменная нам будет необходима для определения, были ли вообще найдены простые множители у заданного числа n. Запоминаем исходное значение переменной n в переменной j (строка 8).

4. В цикле по переменной i начинаем порождение натуральных чисел, не превосходящих середину заданного числа n, для определения делителей данного числа n (строка 9). Данный цикл начали с 2, т.к. единицу мы уже учли (шаг 2 данного алгоритма). Так как в цикле For можно использовать только целые переменные, поэтому воспользовались оператором целочисленного деления на 2 (n div 2). В данном цикле выполняем следующее.

Определяем, является ли очередное i делителем числа n (в качестве n в данном цикле используем j). Для этого определяем остаток от деления j на i. Если остаток равен 0 (строка 10), т.е. число i является делителем j, то определяем, сколько таких делителей, уменьшая число n (строки 11—18). Переменной f присваиваем значение true (строка 12) — это означает, что у заданного числа n есть делители. Организуем цикл, пока остаток от деления j на i равен 0 (строка 13). В данном цикле выводим делитель на экран (строка 15) и уменьшаем заданное число, деля его целочисленно на делитель (строка 16). Повторяем цикл.

После завершения этого цикла возвращаемся на цикл For (строка 9), изменяем i и повторяем те же действия для нового делителя.

5. Если у числа нет делителей (оно является простым), то данное число можно разложить только на 1 и само на себя. Вспомогательная переменная f и определяет, были ли делители у числа n. Если значение переменной f осталось false, то делителей не было, поэтому выводим само это число (строка 19).

```
1)      var i, n, j : integer; f: boolean;
2)      begin
3)      repeat
4)           write('Введите натуральное число N= ');
5)           readln(n);
6)      until n>0;
7)      write (N:6, '=1');
8)      f:=false;   j:=n;
9)      for i:=2 to n div 2 do
10)          if j mod i = 0 then
11)                    begin
12)                      f:=true;
        {цикл определет, сколько таких множителей i в нашем числе n}
13)                    while j mod i=0 do
14)                    begin
```

```
15)                                   write('*', i);
16)                                   j:=j div i;
17)                               end;
18)                           end;
        {f определяет, были ли найдены простые множители,
        большие единицы}
19)       if not f then writeln('*', n);
20)       writeln
21)       end.
```

Задача 1.4. *Даны натуральное n и последовательность* $a_1, a_2, \ldots, a_n$ *вещественных чисел. Найдите знакочередующую сумму* $S = a_1 - a_2 + a_3 - \ldots + (-1)^{n+1} a_n$.

Переменные:
n — количество чисел;
a — очередное число;
p — булевский признак знака слагаемого;
i — переменная цикла;
S — знакочередующая сумма чисел.

Алгоритм решения задачи:

1) вводим длину последовательности n и устанавливаем начальное значение S;
2) булевская переменная p первоначально истинна, она будет указывать на знак слагаемого в сумме;
3) последовательно считываем числа, и если p = true, то прибавляем его к сумме S, иначе отнимаем;
4) на каждом шаге цикла значение p меняем на противоположное;
5) выводим результат.

```
var n, i : integer;
    a, S : real;
    p: boolean;
begin
     repeat
        write('Введите длину последовательности n=');
        readln(n);
    until n>0;
    p:= true;
    S:=0;
    for i:=1 to n do
    begin
        write('Введите a=');
```

```
        readln(a);
        if p then S:=S+a else S:=S–a;
        p:= not p
    end;
    writeln('Знакочередующая сумма  чисел S= ', S);
end.
```

Задача 1.5. *Найти сумму первых n членов ряда* $y = 1 + x/2 + x^2/3 + x^3/4 + \ldots$, *при* $|x|<1$.

Переменные:
n — количество членов ряда;
x — переменная ряда;
z — вспомогательная переменная;
i — переменная цикла;
y — сумма ряда.

Алгоритм решения задачи:
1) вводим количество членов ряда n и переменную x;
2) в цикле порождаем очередной член ряда и прибавляем его к сумме y;
3) выводим результат.

```
var x, y, z : real;
      n, i : integer;
begin
    repeat
        writeln('Введите переменную ряда x, |x|<1, x=');
        readln(x);
        write('Введите число членов ряда n=');
        readln(n);
    until (abs(x)<1) and (n>0);
    y:=1; z:=1;
    for i:=2 to n do
    begin
        z:=z*x;
        y:=y+z/i;
    end;
    writeln('Сумма первых n членов ряда y =', y);
end.
```

Задача 1.6. *Вводится последовательность из N целых чисел. Найти сумму всех отрицательных чисел.*

Переменные:
n — количество чисел;

x — очередное число;
i — переменная цикла;
sum — сумма отрицательных чисел.

Алгоритм решения задачи:

1) вводим длину последовательности n и устанавливаем начальное значение sum;
2) последовательно считываем числа, и если число отрицательное, то прибавляем его к сумме sum;
3) в зависимости от значения sum выводим результат.

```
var n, x, sum, i : integer;
begin
    repeat
       write('Введите длину последовательности n=');
       readln(n);
    until n>0;
    sum:=0;
    for i:=1 to n do
    begin
        write('Введите x='); readln(x);
        if x<0 then sum:=sum+x;
    end;
    if sum=0 then writeln('Отрицательных чисел нет')
             else writeln('Сумма отрицательных чисел sum= ', sum);
end.
```

***Задача 1.7**. Вводится последовательность из N целых чисел. Найти наибольшее число.*

Переменные:
n — количество чисел;
x — очередное число;
i — переменная цикла;
max — наибольшее число.

Алгоритм решения задачи:

1) вводим длину последовательности n и устанавливаем начальное значение max по первому числу;
2) последовательно считываем числа, и если очередное число x больше max, то изменяем значение max := x;
3) выводим результат.

```
var n, x, max, i : integer;
begin
      repeat
         write('Введите длину последовательности n=');
         readln(n);
      until n>0;
      write('Введите x=');
      readln(x);
      max:=x;
      for i:=2 to n do
      begin
         write('Введите x=');
         readln(x);
         if (x>max) then max:=x;
      end;
      writeln('Наибольшее из чисел max=', max);
end.
```

Задача 1.8. *Вводится последовательность целых чисел,* 0 — *конец последовательности. Найти два наименьших числа.*

Переменные:
x — очередное число;
min1 — первое наименьшее число;
min2 — второе наименьшее число (min2>=min1).

Алгоритм решения задачи:

1) устанавливаем начальные значения min1 и min2 по двум первым числам;
2) последовательно считываем числа, и если очередное число x меньше или равно min1 (min1<min2), то изменяем значения min1 и min2;
3) если x попадает в интервал от min1 до min2, то изменяем только min2;
4) выводим результат.

```
var x,min1,min2:integer;
begin
      write('Введите x=');
      readln(x);
      min1:=x;
      write('Введите x=');
      readln(x);
      min2:=x ;
```

```
                                                    {min1<=min2}
    repeat
        if x<=min1 then
                    begin
                        min2:=min1;
                        min1:=x;
                    end
                   else if (min1<x) and (x<min2) then min2:=x;
        write('Введите x=');
        readln(x);
    until (x=0);
    writeln( 'Два наименьших числа равны', min1, 'и', min2);
end.
```

Задача 1.9. *Вводится последовательность ненулевых чисел,* 0 *— конец последовательности. Определить, является ли последовательность возрастающей.*

Переменные:
old — предыдущее число;
new — рассматриваемое число;
f — флаг.

Решение данной задачи строится от противного. Математически для того, чтобы последовательность была возрастающей, для каждого очередного элемента new и предыдущего old должно выполняться условие new > old. Любое нарушение данного условия приводит к тому, что последовательность не может быть возрастающей.

Алгоритм решения задачи:

1) вводим два первых числа как old и new, задаем начальное значение флага f;
2) в цикле ищем нарушение свойства членов возрастающей последовательности;
3) переприсваиваем значение old:=new и вводим новое — new;
4) в зависимости от флага выводим результат.

```
var old, new : real;
            f : boolean;
begin
    write('Введите x=');
    readln(old);
    write('Введите x=');
    readln(new);
```

```
    f:=true;
    repeat
        if new<=old then f:=false;
        old:=new;
        write('Введите x=');
        readln(new);
    until new=0;
    if f then writeln( 'Последовательность возрастающая')
       else writeln( 'Последовательность не является возрастающей');
end.
```

Задача 1.10. *Даны натуральное n и последовательность вещественных чисел a_1, a_2,…, a_n. Сколько отрицательных чисел в начале последовательности (до первого неотрицательного)?*

Переменные:
k — счетчик;
i — переменная цикла;
n — количество членов последовательности;
a — очередной член последовательности;
p — признак отрицательного числа в начале последовательности.

Алгоритм решения задачи:

1) вводим длину последовательности, задаем начальное значение счетчика k;
2) устанавливаем признак отрицательного числа p=true;
3) в цикле вводим очередной член последовательности;
4) если это отрицательное число и до этого неотрицательных чисел не было, то увеличиваем значение счетчика на единицу;
5) в противном случае, если член последовательности неотрицателен, то полагаем p=false;
6) в зависимости от k выводим результат.

```
var a: real; p: boolean;
   k,n : integer;
begin
    repeat
        write('Введите длину последовательности n=');
        readln(n);
    until n>0;
    k:=0; p:=true;
    for i:=1 to n do
    begin
        writeln('Введите число');
```

```
        readln(a);
        if (a<0) and p then k:=k+1else
        if a>=0 then p :=false
    end;
    if k=0 then writeln('отрицательных чисел  в начале нет')
           else writeln('последовательность начинается с ', k, ' чисел')
end.
```

Задача 1.11. *Дан прямоугольный бильярдный стол со сторонами A и B, где A, B — натуральные числа (бильярд Льюиса Кэрролла). Из угловой лузы вылетает шар под углом 45 градусов к боковым стенкам, ударяется о борт, отскакивает, ударяется еще раз и т.д., пока не вылетит через одну из угловых луз. Рассчитать количество отрезков в ломаной траектории шара. Считать угол падения равным углу отражения.*

Данная задача решается с помощью стандартных функций выделения целой части от деления y на x (y div x) и выделения остатка y mod x. При прохождении шаром прямоугольного стола и отражении его от боковых сторон происходит увеличение числа отрезков траектории на два, а обратный путь вычисляется как y:=a–x+y mod x, где y — обратный путь для шара, a — длинная сторона стола, x — короткая сторона стола.

Переменные:
а) в функции bill:
x, y — два натуральных числа (формальные параметры);
k — вспомогательная переменная (локальная переменная);
a — длинная сторона стола (глобальная переменная);
б) в основной программе:
a, b — два натуральных числа (глобальные переменные).

Алгоритм решения задачи:
1) создаем описание функции bill;
2) вводим два натуральных числа a и b (не кратные друг другу);
3) вызываем функцию bill для определения количества отрезков;
4) завершаем работу программы.

```
var a, b : integer;
function bill(y,x:integer):integer;
  var k:integer;
  begin
      k:=0;
      while y mod x <>0 do
      begin
```

```
            k:=k+y div x+2;
            y:=a–x+y mod x;
        end;
  bill:=k;
  end;

begin
     repeat
         writeln('Введите два натуральных числа A>B');
          readln(a,b);
     until a>=b;
     writeln('Количество отрезков в траектории :', bill(a,b));
end.
```

Задача 1.12. *Пусть процедура maxmin(x,y) присваивает параметру x большее из вещественных чисел x и y, а параметру y — меньшее. Описать данную процедуру и использовать ее для перераспределения значений вещественных переменных a, b и c так, чтобы стало a>=b>=c.*

```
var a,b,c : real;
procedure maxmin( var x,y:real);
  var r:real;
   begin if x<y then begin r:=x; x:=y; y:=r end   end;

begin
     writeln('Введите три числа a,b,c –');
     readln(a,b,c);
     maxmin(a,b);
     maxmin(a,c);                                  {a=max}
     maxmin(b,c);                                 {c=min}
     writeln(a,b,c);
 end.
```

Задача 1.13. *Если среди чисел sin(x ^{n}) (n = 1, 2, ... ,30) есть хотя бы одно отрицательное число, то логической переменной t присвоить значение true, а иначе — значение false.*

```
var y,x : real;
       n : integer;
       t : boolean;
begin
      write('Введите значение x –');
      readln(x);
```

```
    y:=1; n:=0;
    repeat
        n:=n+1; y:=x*y; t:=sin(y)<0
    until t or (n=30);
    writeln(t);
end.
```

Задача 1.14. *Определить k — количество трехзначных натуральных чисел, сумма цифр которых равна n (1<n<27). Операции деления (/, div и mod) не использовать.*

```
var d1, d2, d3, k, n : integer;
begin
    writeln('Введите число n, с которым будем сравнивать сумму цифр
числа');
    readln(n);
    k:=0;
    {d1 – левая, d2 – средняя, d3 – правая цифры числа}
    for d1:=1 to 9 do
      for d2:=0 to 9 do
        for d3:=0 to 9 do
          if d1+d2+d3=n then begin
                                k:=k+1; write(d1,d2,d3, ' ');
                              end;
     writeln('Количество искомых чисел равно –', k);
end.
```

Строковый тип данных

Задача 1.15. *Вывести в одну строку ABBCCCDDDDE .... ZZ...Z.*

Переменные:
i — переменная цикла; определяет, какая буква выводится;
k — количество повторений буквы;
j — переменная цикла.

Алгоритм решения задачи:
1) цикл for i:='A' to 'z' do определяет, какую букву выводим на печать;
2) внутренний цикл for j:=1 to k do определяет, сколько раз будет печататься буква;
3) выводим заданную букву k раз на экран;
4) после вывода всех букв закрываем строку оператором writeln.

```
var   i : char;
    k,j : integer;
begin
       k:=1;
       for i:= 'A' to 'Z' do
       begin
            for j:=1 to k do  write(i);
             k:=k+1;
       end;
       writeln;
end.
```

Задача 1.16. *Дана строка символов. Удалить из нее все знаки препинания.*

Переменные:
i — переменная цикла;
L — длина строки;
str — строка текста;
str1 — вспомогательная строка;
m — множество знаков препинания.

Алгоритм решения задачи:

1) задаем значение множества m — множества знаков препинания;
2) вводим строку str с клавиатуры;
3) цикл for i:=1 to l do осуществляет построение вспомогательной строки без знаков препинания: берем отдельный символ строки и проверяем, является ли он знаком препинания. Если да, то присоединяем этот символ к новой вспомогательной строке. Если нет, то переходим к следующему элементу строки;
4) первоначальную строку заменяем на вспомогательную;
5) выводим сроку на экран.

```
var   str, str1 : string;
      L, i : integer;
       m : set of char;
begin
       m:=['.', ',', '!', ':', ';', '?', '-'];
       writeln('Введите текст');
       readln(str);  L:=length(str); str1:='';
       for i:=1 to L do
            if not(str[i] in m) then str1:=str1+str[i];
       str:=str1;
       writeln('Преобразованный текст ', str);
end.
```

***Задача 1.17**. Дана строка символов. Выделить подстроку между первой и последней точкой.*

Переменные:
i — номер позиции, которая соответствует точке;
j — местоположение первой точки;
m — местоположение последней точки;
str — строка текста;
s1 — вспомогательная переменная.

Алгоритм решения задачи:

1) вводим строку str и присваиваем значение вспомогательной переменной s1:=str;
2) определяем местоположение первой точки в тексте; если точка есть, то вырезаем из s1 текст до нее;
3) ищем последнюю точку в цикле while i<>0 do; если она есть, то значение переменной m указывает на ее местоположение;
4) в зависимости от присутствия точек выделяем подстроку s1:=copy(str,j+1,m–j–1) и выводим результат на экран.

```
var str, s1 : string;
    i, m, j : integer;
begin
     writeln('Введите текст');
     readln(str);
     s1:=str; i:=pos('.',s1); j:=i; m:=0;
     if (i<>0) then
                begin
                  while i<>0 do
                      begin delete(s1,1,i); m:=m+i; i:=pos('.',s1); end;
                   if m<>j then
                             begin
                              s1:=copy(str,j+1,m–j–1);
                              writeln('Часть текста между 1-й и последней точкой');
                              writeln(s1);
                              end
                     else writeln('В тексте только одна точка');
                end
                else writeln('В тексте нет ни одной точки');
end.
```

***Задача 1.18**. Дана строка символов. Определить, является ли она записью десятичного числа, кратного трем.*

Необходимо удостовериться, что введенная строка состоит только из цифр, т.е. может быть преобразована в десятичное число. Само преобразо-

вание цифры в число сделать очень просто. Поскольку коды символьных переменных — цифр — следуют один за другим, то функция ord в выражении ord(s1[i])–ord('0') поможет нам сделать это.

Переменные:
i — переменная цикла;
s1 — строка цифр;
m — длина строчки;
k — счетчик цифр.

Алгоритм решения задачи:

1) вводим строку s1;
2) организуем в строке поиск цифр до тех пор, пока не встретим конец строчки или не обнаружим наличие ошибки в арифметическом выражении;
3) если встречается цифра, то добавляем ее к общей сумме цифр;
4) после определения суммы цифр, определяем, кратна ли она трем;
5) в зависимости от полученного результата выводим ответ на экран.

```
var  s1 : string;
      k, i, m : integer;
begin
      writeln('Введите строку');
      readln(s1);
      m:=length(s1);
      k:=0; i:=1;
      repeat
            case s1[i] of
            '0'..'9' :  k:=k+(ord(s1[i])–ord('0'));
            '  ' : ;
            else k:=–1;
            end;
            i:=i+1;
      until (i>m) or (k<0);
      if k mod 3 = 0
                  then writeln('Это десятичное число, кратное 3')
                  else writeln('Это не десятичное число, кратное 3')
end.
```

Задача 1.19*. Дана строка символов. Группы символов в ней между группами пробелов считаются словами. Посчитать, сколько слов содержит данная строка.*

Переменные:

i — вспомогательная переменная;

s — строка текста;

k — счетчик слов.

Алгоритм решения задачи:

1) вводим строку s;

2) на каждом шаге внешнего цикла отыскиваем очередное слово в строке и увеличиваем счетчик слов;

3) выводим результат на экран.

```
var s: string;  i,k: integer;
begin
  writeln('Введите строку');
  readln(s);
  k:=0; i:=1;
  while i<=length(s) do
      begin
      {пропускаем пробелы}
      while (s[i]=' ') and (i<=length(s)) do i := i+1;
      if i<=length(s) then k := k+1;
      {ищем новый пробел}
      while (s[i]<>' ') and (i<=length(s)) do i := i+1;
      end;
  writeln('количество слов =',k)
end.
```

Задача 1.20. *Дана строка символов. Группы символов в ней между группами пробелов считаются словами. Определить длину самого короткого и самого длинного слова.*

Переменные:

i — индекс очередного символа в строке;

s — строка текста;

beginStr и endStr — начало и конец слова;

len — длина слова;

max — длина наибольшего слова;

min — длина наименьшего слова.

Алгоритм решения задачи:

1) вводим строку s;

2) устанавливаем начальные значения max и min;

3) организуем внешний цикл для поиска очередного слова в строке;

4) найдя слово, определяем его длину и при необходимости корректируем max и min;

5) выводим результат на экран.

```
var s:string;
   i,min,max,beginStr,endStr,len:integer;
begin
  writeln('Введите строку');
  readln(s);
  max:=0; min:=255;
  i:=1;
  while i<=length(s) do
      begin
      {пропускаем пробелы}
      while (s[i]=' ') and (i<=length(s)) do i:=i+1;
      if i<=length(s) then begin
                        beginStr:=i;
                        {нашли начало слова, ищем его конец}
                        while (s[i]<>' ') and (i<=length(s)) do
                                i:=i+1;
                        endStr:=i–1;
                        writeln(beginStr,'–',endStr);
                        len:=endStr–beginStr+1;
                        if len>max then max:=len;
                        if len<min then min:=len;
                        end
      end;
  writeln('длина самого длинного слова =',max);
  writeln('длина самого короткого слова =',min)
end.
```

Задача 1.21. *Дана строка. Преобразовать строку, заменив в ней каждую группу стоящих рядом точек одной точкой.*

Переменные:

s — строка текста.

Для решения задачи:

1) вводим строку s;

2) найдем пару стоящих рядом точек, удалим одну из них; повторим поиск. И так до тех пор, пока в строке не окажется ни одной такой пары точек;

3) выводим результат на экран.

```
var s : string;
begin
      writeln('Введите строку');
      readln(s);
      while pos('..', s) <> 0 do delete (s, pos('..', s), 1);
      writeln('Полученная строка : ', s);
end.
```

Задача 1.22. *Присвоить литерным переменным c2, c1 и c0 соответственно левую, среднюю и правую цифры трехзначного числа k.*

```
var c0, c1, c2 : char;
       n0, k, d : integer;
begin
      writeln('Введите трехзначное число');
      readln(k);
      n0:=ord('0');
      d:=k div 100;              c2:=chr(n0+d);
      d:=k mod 100 div 10; c1:=chr(n0+d);
      d:=k mod 10;               c0:=chr(n0+d);
      writeln('c2= ', c2);
      writeln('c1= ',c1);
      writeln('c0= ',c0);
end.
```

Задача 1.23. *Используя только литерный ввод, т.е. процедуру readln(c), где c — литерная переменная, ввести непустую последовательность цифр, перед которой может находиться знак «+» или «–» и за которой следует пробел, и, получив соответствующее число, присвоить его целой переменной k.*

```
var c : char;
    sign, n0, k : integer;
begin
       {определение знака числа:}
       sign:=1;
       readln(c);
       if c='–' then
              begin sign:=–1; readln(c) end
              else    if c='+' then readln(c);
       {чтение цифр (первая – в c) и вычисление абсолютной величины числа k по схеме Горнера:}
       n0:=ord('0'); k:=0;
```

```
        repeat
              k:=10*k+ord(c)–n0;
               readln(c)
        until c=' ';
        {учет знака:}
        k:=sign*k;  writeln(k);
end.
```

Задача 1.24. *Программа. Напечатать заданную непустую строку:*
а) удалив из нее все цифры и удвоив знаки «+» и «–»:

```
var s : string;
     i : integer;
begin
        writeln('Введите строку:');
        readln(s);
        for i:=1 to length(s) do
              if (s[i]='+')or(s[i]='–') then write(s[i],s[i])
                                    else  if (s[i]<'0')or(s[i]>'9') then write(s[i]);
        writeln
end.
```

б) удалив из нее все знаки «+», непосредственно за которыми идет цифра:

```
var a, b : char;    {a – очередная литера строки, b – следующая}
        s : string;
         i : integer;
begin
       writeln('Введите строку:');
       readln(s);
       for i:=1 to length(s) do
       begin
            a:=s[i];
            if a<>'+' then write(a)
                       else   if i<length(s) then begin
                                                    b:=s[i+1];
                                                    if (b<'0') or (b>'9') then write(a)
                                                    end
                                              else write(a);
       end;
       writeln
end.
```

Варианты заданий лабораторной работы № 1

Вариант 1	Задача 1 Определить число, получаемое выписыванием в обратном порядке цифр заданного натурального числа n. Вводите число n как значение типа integer. Например, если n = 1234, то ответ равен 4321. Строки не использовать. Подсказка. Как получать цифры целого числа, см. учебное пособие В.М. Зюзькова «Программирование», раздел 3.5. Задача 2 Для заданной строки определить длину содержащейся в ней максимальной подстроки, не имеющей латинских букв.
Вариант 2	Задача 1 Даны целое n>2 и вещественные числа a_1, b_1, ..., a_n, b_n ($a_i < b_i$). Рассматривая пары a_i и b_i как левые и правые концы отрезков на одной и той же прямой, определить концы отрезка, являющегося пересечением всех этих отрезков. Если такого отрезка нет, сообщить об этом. Например, если n = 3 и $a_1 = 0$, $b_1 = 10$, $a_2 = 1$, $b_2 = 11$, $a_3 = –1$, $b_3 = 5$, то результатом будет отрезок [1, 5]. Задача 2 Дана строка S. Создать новые строки из строки S: а) заменить все восклицательные знаки точками; б) заменить каждую точку многоточием (т.е. тремя точками); в) заменить каждую из групп стоящих рядом точек одной точкой; г) заменить каждую из групп стоящих рядом точек многоточием (т.е. тремя точками).
Вариант 3	Задача 1 Не используя стандартные функции, вычислить с точностью eps > 0 $$y = \cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots + \frac{(-1)^n x^{2n}}{(2n)!} + \ldots$$ Считать, что требуемая точность достигнута, если очередное слагаемое по модулю меньше eps; все последующие слагаемые можно уже не учитывать. Вложенные циклы не использовать. Подсказка: в двух разных переменных храните отдельно числитель и знаменатель очередного слагаемого и на каж-

	дом шаге вычисляйте новые числитель и знаменатель через предыдущие значения. Задача 2 В заданную непустую строку входят только цифры и буквы. Определить, удовлетворяет ли строка следующему свойству: строка содержит (помимо букв) только одну цифру, причем ее числовое значение равно длине строки.
Вариант 4	Задача 1 Даны натуральные числа n, m. Получить сумму m последних цифр числа n. Число n вводить как величину типа integer, и строки не использовать. Пример. Пусть n = 12345, m = 3, тогда ответ равен 3 + 4 + 5 = = 12. Подсказка. Как получать цифры целого числа, см. учебное пособие В.М. Зюзькова «Программирование», раздел 3.5. Задача 2 Для каждой цифры '0', '1', ... , '9' подсчитать количество вхождений в данную строку S.
Вариант 5	Задача 1 Определить функцию f(n), значением которой является натуральное число, получаемое выбрасыванием из записи натурального числа n первой справа цифры 0 или 5. Например, f(13510) = 1351, f(1351) = 131. Для данного натурального числа m, примените функцию f последовательно необходимое число раз, так чтобы в записи m не осталось цифр 0 и 5. Подсказка. Как получать цифры целого числа, см. учебное пособие В.М. Зюзькова «Программирование», раздел 3.5. Задача 2 Составьте программу, в результате выполнения которой в первой заданной строке удваивается каждый символ, принадлежащий также второй строке.
Вариант 6	Задача 1 Дано натуральное n > 0. Найти произведение первых n простых чисел. Подсказка: используйте булевскую функцию для проверки, является число простым числом или нет. Задача 2 Из данной строки удалить все цифры и малые латинские буквы.

Вариант 7	Задача 1 Даны длины a, b и c сторон некоторого треугольника. Найти медианы треугольника, сторонами которого являются медианы исходного треугольника. Замечание: длина медианы, проведенной к стороне a, равна $0{,}5 * \sqrt{2*b^2 + 2*c^2 - a^2}$. Определите функцию для вычисления медианы и используйте ее необходимое число раз. Задача 2 Ввести две строки S1 и S2, состоящие только из цифр. Определить, в какой строке сумма цифр большая.
Вариант 8	Задача 1 Дано: натуральное n, действительные a_1, a_2,...,a_n. Вычислить: $a_1 + a_2 * (a_2 - 1) + a_3 * (a_3 - 1) * (a_3 - 2) + a_4 * (a_4 - 1) * * (a_4 - 2) * (a_4 - 3) + ... + a_n * (a_n - 1) * (a_n - 2) *...* (a_n - n + 1)$. Задача 2 В данной строке найти самую длинную подстроку, состоящую из одинаковых символов.
Вариант 9	Задача 1 Пусть n — натуральное число. Обозначим через n!! произведение 1 * 3 * ... * n для нечетного n и 2 * 4 * 6 * ...* n для четного n. Дано натуральное n. Получить n!!. Задача 2 Дана строка S, и дано натуральное n. Удалить из строки S все группы длиной n подряд стоящих одинаковых символов. Указание: если ввести строку, например ‘kjhdahhhasaaaaa’, а значение n ввести равное 2, то необходимо получить ‘kjhdahasaaa’.
Вариант 10	Задача 1 Дано n вещественных чисел. Найти порядковый номер того из них, которое наиболее близко к квадрату какого-нибудь целого числа. Например, если n = 10 и вводим числа 110, 90, 80, 30, 50, 40, 40, 30, 22, 60, то ответ равен 3 или 5 (имеем $\|80 - 9^2\| = \| 50 - 7^2\| = 1$). Подсказка: определите функцию f(k), которая вычисляет абсолютную величину разности k и ближайшего целого квадрата (ближайшим целым квадратом будет $(\text{trunc}(\sqrt{k}))^2$ или $(\text{trunc}(\sqrt{k}) + 1)^2$).

	Задача 2 Для заданной строки определить длину содержащейся в ней максимальной подстроки, не имеющей латинских букв.
Вариант 11	Задача 1 Вычислить бесконечную сумму с заданной точностью E (E > 0). $$\sum_{i=0}^{\infty} \frac{(-1)^i}{i!}.$$ Считать, что требуемая точность достигнута, если вычислена сумма нескольких первых слагаемых и очередное слагаемое оказалось по модулю меньше, чем E; это и все последующие слагаемые можно уже не учитывать. Вложенные циклы не использовать. Подсказка: вместо вычисления факториала вычисляйте величину, обратную факториалу, тем самым вы избежите целочисленного переполнения (см. учебное пособие В.М. Зюзькова «Программирование», раздел 3.6, задача 3). Задача 2 Дана строка S. Каждую подстроку длиной n > 1, состоящую из одинаковых символов, заменить на подстроку из тех же символов в количестве 2*n штук. Например, дана строка ‘kjhhsdaaa’. Необходимо получить и вывести на зкран строку: ‘kjhhhhsdaaaaaa’.
Вариант 12	Задача 1 Числа Фибоначчи: $F_0 = 0$, $F_1 = 1$, а любое следующее число Фибоначчи равно сумме двух предыдущих: $F_n = F_{n-1} + F_{n-2}$. Известно, что при достаточно больших n справедливо приближенное равенство $$F_n = \frac{1}{\sqrt{5}} * \left(\frac{1+\sqrt{5}}{2} \right)^n.$$ Определите наименьший номер n, начиная с которого равенство выполняется с точностью до заданного eps. Задача 2 Определить, является ли заданная строка правильной записью вещественного числа (возможно, со знаком, а также с пробелами спереди и/или сзади). Указание: вещественное число может быть представлено либо в виде числа с фиксированной десятичной точкой (–214.78), либо в виде числа с плавающей десятичной точкой (–2.1478E+02).

Вариант 13	Задача 1 Рассмотрим последовательность e_1, e_2, e_3,..., образованную по следующему закону $$e_k = \left(1+\frac{1}{k}\right)^k, \quad k = 1,2,....$$ Дано действительное eps. Найти первый член e_n этой последовательности, для которого $\lvert e_n - e_{n-1}\rvert < eps$. (Существование e_n гарантируется одной из теорем математического анализа: чем меньше eps, тем ближе e_n к числу e = = 2,718281828... .) Задача 2 В заданном предложении найти пару слов, из которых одно является обращением другого. Указание: воспользуйтесь массивом строк для хранения слов.
Вариант 14	Задача 1 Даны натуральные n, m. Получить все меньшие n натуральные числа, сумма цифр которых равна m. Подсказка. Как получать цифры целого числа, см. учебное пособие В.М. Зюзькова «Программирование», раздел 3.5. Задача 2 Проверить, правильно ли в данной строке расставлены круглые скобки (т.е. находится ли справа от каждой открывающей скобки соответствующая закрывающая скобка, а слева от каждой закрывающей — соответствующая открывающая). Ответ — «да» или «нет». Используйте следующий алгоритм: На каждом шаге цикла во время просмотра строки символов текущее значение количества открывающих скобок больше или равно текущему значению количества закрывающих скобок. После окончания цикла количество открывающих скобок равно количеству закрывающих скобок.
Вариант 15	Задача 1 Пусть дано натуральное число n. Составить программу вычисления n^3 как суммы нечетных чисел исходя из того, что: $1^3 = 1$; $2^3 = 3 + 5$; $3^3 = 7 + 9 + 11$; $4^3 = 13 + 15 + 17 + 19$; $5^3 = 21 + 23 + 25 + 27 + 29$; …

	Задача 2 Исключить из данной строки группы символов, расположенные между скобками (,). Сами скобки тоже должны быть исключены. Указание: предполагается, что внутри каждой пары скобок нет других скобок.
Вариант 16	Задача 1 Вычислить с точностью eps>0 бесконечную сумму $$1+\frac{1}{3^4}+\frac{1}{5^4}+\frac{1}{7^4}+...$$ Считать, что требуемая точность достигнута, если очередное слагаемое по модулю меньше eps; все последующие слагаемые можно уже не учитывать. Задача 2 Даны две строки S1 и S2. Создать строку, состоящую из символов, входящих либо в S1, либо в S2, но не в обе сразу.
Вариант 17	Задача 1 Вычислить с точностью eps>0 бесконечную сумму $$\frac{1}{1\cdot 3}+\frac{1}{3\cdot 5}+\frac{1}{5\cdot 7}+...$$ Считать, что требуемая точность достигнута, если очередное слагаемое по модулю меньше eps; все последующие слагаемые можно уже не учитывать. Задача 2 Даны две строки S1 и S2. Создать строку, состоящую из латинских букв, не принадлежащих какой-либо строке S1 и S2. Указание: результирующая строка должна содержать те латинские буквы, которых вообще нет ни в строке S1, ни в строке S2.
Вариант 18	Задача 1 Составить программу вычисления значения функции $$Z(x)=\cfrac{x}{1+\cfrac{x}{2+\cfrac{x}{3+\cfrac{x}{\cdot\ \cdot\ \cdot\\ \cdot\ \cdot\ \cdot\\ 100+\cfrac{x}{101}}}}}$$

	Задача 2 Для заданных трех строк S1, S2 и S3 напечатать те латинские буквы, которые входят только в одну из этих строк. Указание: например, ввели строки S1 — ‘asfgsas’, S2 — ‘qweasdyu’, S3 — ‘mnbqwef’. Ответом должно служить следующее: ‘Только в первую строку входят буквы — g’, ‘Только во вторую строку входят буквы — dyu’, ‘Только в третью строку входят буквы — mnb’.
Вариант 19	Задача 1 Вводим вещественные числа x_1, x_2, … до тех пор, пока не введем отрицательное число (первое число – неотрицательное). И пусть x_1, x_2, …, x_n — члены данной последовательности, предшествующие отрицательному числу. Посчитать количество полных квадратов среди чисел x_1, x_2, …, x_n. Подсказка. Число k является полным квадратом, если выполнено равенство $k = (\text{trunc}(\sqrt{k}))^2$. Задача 2 Для заданных трех строк S1, S2 и S3 определить, какая из них является десятичной записью числа, кратного 3. Указание: строка может быть любой длины, поэтому переводить строку целиком в число не рекомендуется. Вспомните и используйте другой метод определения кратности числа 3.
Вариант 20	Задача 1 Даны натуральное число n, действительные числа x_1, x_2, …, x_n. Найти длину наименьшего отрезка числовой оси, содержащего все числа x_1, x_2, …, x_n. Задача 2 Для заданных трех строк S1, S2 и S3 определить, какая из них является палиндромом. Назовем строку палиндромом, если его запись читается одинаково с начала и с конца (как, например, ‘казак’, ‘алла’, ‘fffjjjfff’). Указание: определить булевскую функцию f(s) для проверки, является ли строка s палиндромом. Результатом должен быть ответ «да» или «нет».

ЛАБОРАТОРНАЯ РАБОТА № 2

Лабораторная работа № 2 посвящена созданию программ с использованием массивов (одномерных и матриц) и множеств. В этих программах полезно использовать подпрограммы.

Задание состоит из двух задач, требующих написания программ на языке Паскаль. Программы должны быть написаны в рамках структурного программирования. В частности, запрещается использовать операторы перехода и метки.

Примеры решения задач

Одномерные массивы

Задача 2.1. *Дан массив чисел. Найти, сколько в нем пар одинаковых соседних элементов.*

Переменные:
mas — массив чисел;
n — размер массива;
i — переменная цикла;
k — количество одинаковых пар соседних элементов.

Алгоритм решения задачи:

1) вводим длину массива n, значение элементов массива и устанавливаем начальное значение k;
2) последовательно просматриваем элементы, и если очередной mas[i] равен следующему mas[i+1], то увеличиваем значение k на единицу;
3) выводим результат.

```
const m=100;
var mas : array [1..m] of integer;
      i, k, n : integer;
begin
    write('Введите размер массива n=');
    readln(n);
    k:=0;
    for i:=1 to n do
    begin
        write('Введите элемент массива');
        readln(mas[i]);
    end;
    for i:=1 to n–1 do
```

```
    if mas[i]=mas[i+1] then k:=k+1;
    wrietln('Одинаковых пар соседних элементов ',k);
end.
```

Задача 2.2. *Программа. Дано 100 целых чисел. Распечатать их в обратном порядке по 6 чисел в строке.*

```
const n=100; d=6;
var x : array [1..n] of integer;
    i, k : integer;
begin
      {ввод массива:}
      writeln('Введите массив из ', n,' чисел');
      for i:=1 to n do read(x[i]);
       {вывод массива по d чисел в строке:}
      k:=0;              {номер числа в строке}
      for i:=n downto 1 do
      begin
           k:=k+1;
           write(x[i]:4);
           if k=d then
                  begin
                  k:=0; writeln
                  end
      end;
      if k<>0 then writeln
end.
```

Задача 2.3. *Сортировка слиянием*

```
const k=50; m=20; n=70; {n=k+m}
var x : array [1..k] of real;
     y : array [1..m] of real;
     z : array [1..n] of real;
     i,p,j : integer;
```

Элементы каждого из массивов x и y упорядочены по неубыванию. Объединить элементы этих двух массивов в один массив z так, чтобы они снова оказались упорядоченными по неубыванию.

```
const k=50; m=20; n=70;                                    {n=k+m}
var x : array [1..k] of real;
    y : array [1..m] of real;
    z : array [1..n] of real;
```

```
    i, p, j : integer;
begin
    writeln('Введите массив x');
    for i:=1 to k do read(x[i]);
    writeln;
    writeln('Введите массив y');
    for i:=1 to m do read(y[i]);
    writeln;
    writeln('Введенные Вами массивы');
    writeln('Массив x');
    for i:=1 to k do write(x[i]:7:3);
    writeln;
    writeln('Массив y');
    for i:=1 to m do write(y[i]:7:3);
    writeln;
    p:=1;                {индекс очередного элемента из z}
    i:=1;                {из x}
    j:=1;                {из y}
     {пока есть нерассмотренные элементы и в x, и в y: }
    repeat if x[i]<y[j] then
                         begin
                             z[p]:=x[i]; i:=i+1
                         end
                         else
                         begin
                            z[p]:=y[j]; j:=j+1
                         end;
              p:=p+1
    until (i>k)or(j>m);
    {один из массивов x или y исчерпан --> перепись в z
    оставшихся элементов другого массива:}
    if i>k               {исчерпан массив x:}
           then repeat z[p]:=y[j]; j:=j+1; p:=p+1 until j>m
           else repeat z[p]:=x[i]; i:=i+1; p:=p+1 until i>k;
    writeln('Полученный массив: ');
    for i:=1 to n do write(z[i]:7:3);
end.
```

Задача 2.4. *type shift=1..99;*
scale=array [1..100] of real;

Описать процедуру move(s,k), которая преобразует шкалу s, циклически сдвигая ее элементы на k позиций влево, где k — параметр типа shift.

```
type shift=1..99;
      scale=array [1..100] of real;
var a:scale; m:shift;n,i:integer;
procedure move( var s:scale; k:shift);
  var i:integer; t:scale;                    {вспомогательный  массив}
   begin
                                                {t[n–k+1..n]:=s[1..k]:}
      for i:=1 to k do t[n–k+i]:=s[i];
                                                {t[1..n–k]:=s[k+1..n]:}
      for i:=k+1 to n do t[i–k]:=s[i];
      s:=t
   end;

begin
     writeln('Введите размер массива');
     readln(n);
     writeln('Введите массив');
     for i:=1 to n do read(a[i]);
     writeln('Введите цикл сдвига <=', n–1);
     readln(m);
     writeln('Исходный массив');
     for i:=1 to n do write(a[i]:7:2);
     writeln;
     move(a,m);
     writeln('Полученный массив');
     for i:=1 to n do write(a[i]:7:2);
     writeln;
end.
```

Многомерные массивы

Задача 2.5*. Дана матрица N M, состоящая из натуральных чисел. Выбрать в строках самые левые наименьшие элементы и поставить их в первый столбец.*

Для решения этой задачи нужно сначала найти самый левый минимальный элемент в каждой строке и запомнить его местоположение, а затем поменять его местами с элементом в первом столбце.

Переменные:
a — двумерный массив;
n, m — количество строк и столбцов массива;
i, j — переменные цикла;
jm — столбец минимального элемента для каждой строки;

min — текущий минимум.

Алгоритм решения задачи:

1) вводим размеры массива A и значения его элементов;
2) просматриваем строки массива слева направо, ищем минимальное значение и запоминаем значения индексов;
3) для каждой строки меняем местами минимальный элемент и элемент в первом столбце;
4) выводим матрицу на экран.

```
const t=100; s=100;
var a : array [1..t,1..s] of integer;
   n, m, jm, i, j, min : integer;
begin
   write('Введите количество строк n=');
   readln(n);
   write('Введите количество столбцов m=');
   readln(m);
   for i:=1 to n do
     begin
       write('Введите через пробел', m,'чисел');
       for j:=1 to m do
       read(a[i,j]);
     end;
  for i:=1 to n do
  begin
     min:= a[i,1];
     jm:=1;
       for j:=1 to m do
       if min > a[i,j]
            then
                begin
                jm:=j;
                min:=a[i,j];
                end;
       a[i,jm]:=a[i,1];
       a[i,1]:=min;
  end;
  for i:=1 to n do          {вывод матрицы на экран в виде таблицы}
  begin
     for j:=1 to m do
     write(a[i,j]:4);
     writeln;
  end
```

end.

Задача 2.6. *Дана квадратная матрица N×N, состоящая из натуральных чисел. Зеркально отразить ее элементы относительно побочной диагонали. Вывести результат на экран.*

Рассмотрим матрицу 3×3 и посмотрим, что происходит с элементами при зеркальном отображении:

```
A11 A12 A13          A33 A23 A13
A21 A22 A23   <->    A32 A22 A12
A31 A32 A33          A31 A21 A11
```

Если считать, что после преобразования у нас появилась новая матрица B, то соответствие между элементами устанавливается следующим образом:

B11 <-> A33
B12 <-> A23
B21 <-> A32
B22 <-> A22 и т.д., т.е. B[I,J] <-> A[L,M]

Внимательно изучив соответствие, можно утверждать, что для элементов матрицы NxM справедлива следующая система уравнений:

I+M = N+1,
J+L = N+1.

Отсюда правило преобразования элементов выглядит следующим образом:

B[I,J] = A[N+1–J,N+1–I].

Переменные:

a, b — двумерные массивы;
m — количество строк и столбцов массива;
i, j — переменные цикла;
k — вспомогательная переменная.

Алгоритм решения задачи:

1) вводим размеры массива A и присваиваем значения его элементам;
2) присваиваем значения элементам матрицы B по представленным выше формулам и выводим их на экран.

Программа, решающая данную задачу, выглядит так:

```
const n=100;
var a,b : array [1..n] of integer;
   k, m, i, j : integer;
begin
```

```
      write('Введите размер матрицы m=');
      readln(m);
      writeln('Исходная матрица');
      k:=1;
      for i:=1 to m do
          for j:=1 to m do
           begin
              a[i,j]:=k; k:=k+1;
              if j<m then write(a[i,j]:4)
                        else writeln(a[i,j]:4)
           end;
      writeln('Матрица после преобразования');
      for i:=1 to m do
          for j:=1 to m do
          begin
               b[i,j]:= a[m+1–j, m+1–i];
               if j<m then write(b[i,j]:4)
                         else writeln(b[i,j]:4)
          end;
end.
```

Задача 2.7. *Вычислить: а) C = A + B*

Создадим две процедуры:

tab_in(var a1 : mas) — для ввода элементов массива с клавиатуры. Параметр a1 — формируемый массив;

tab_out(var b1 : mas) — для вывода массива на экран. Параметр b1 — имя выводимого массива.

```
type mas=array [1..40,1..40] of real;
      vec=array[1..40] of real;
var a,b,c : mas;
     x,y : vec;
     i, j, n, k : integer;
     r, s : real;
procedure tab_in(var a1 : mas);
var i, j : integer;
begin
    for i:=1 to n do
       for j:=1 to n do
       read(a1[i,j]);
    writeln;
end;
```

```
procedure tab_out(var b1 : mas);
var i,j:integer;
begin
    for i:=1 to n do
    begin
       for j:=1 to n do
       write(b1[i,j]:7:3);
    writeln;
    end;
end;

begin
      writeln('Введите размер матрицы');
      readln(n);
      writeln('Введите матрицу A');
      tab_in(a);
      writeln('Введите матрицу B');
      tab_in(b);
      writeln('Матрица A');
      tab_out(a);
      writeln('Матрица B');
      tab_out(b);
      for i:=1 to n do
         for j:=1 to n do
            c[i,j]:=a[i,j]+b[i,j];
      writeln('Полученная матрица C=A+B');
      tab_out(c);
end.
```

*б) y = A * x*

Основная программа

```
begin
     writeln('Введите размер матрицы');
     readln(n);
     writeln('Введите матрицу A');
     tab_in(a);
     writeln('Введите вектор x');
     for i:=1 to n do read(x[i]);
     writeln;
     writeln('Матрица A');
     tab_out(a);
```

```
    writeln('Введенный вектор x');
    for i:=1 to n do write(x[i]:7:2);
    writeln;
    for i:=1 to n do
    begin
        s:=0;
        for j:=1 to n do
                s:=s+A[i,j]*x[j];
        y[i]:=s
    end;
    writeln('Полученный вектор y=A*x');
    for i:=1 to n do
    write (y[i]:7:3);
    writeln;
end.
```

*в) C = A * B*

Основная программа

```
begin
     writeln('Введите размер матрицы');
      readln(n);
      writeln('Введите матрицу A');
      tab_in(a);
      writeln('Введите матрицу B');
      tab_in(b);
      writeln('Матрица A');
      tab_out(a);
       writeln('Матрица B');
      tab_out(b);
      for i:=1 to n do
          for j:=1 to n do
          begin
                s:=0;
                for k:=1 to n do
                      s:=s+a[i,k]*b[k,j];
                c[i,j]:=s;
          end;
      writeln('Полученная матрица C=A+B');
      tab_out(c);
end.
```

г) транспонированную матрицу B

Основная программа

```
begin
    writeln('Введите размер матрицы');   readln(n);
    writeln('Введите матрицу B'); tab_in(b);
    writeln('Матрица B'); tab_out(b);
    for i:=1 to n–1 do
        for j:=i+1 to n do
        begin
            r:=b[i,j]; b[i,j]:=b[j,i]; b[j,i]:=r;
        end;
    writeln;
    writeln('Полученная транспонированная матрица B');
    tab_out(b);
end.
```

Множества

Задача 2.8*. type str=string[100];*

Описать функцию count(s), подсчитывающую общее количество цифр и знаков '+', '–' и '', входящих в строку s.*

Переменные:

а) в функции count:

s — строка (формальный параметр);

i — счетчик цикла (локальная переменная);

k — общее количество цифр и знаков ‘+’, ‘*’, ‘–‘ в заданной строке (локальная переменная);

б) в основной программе:

s1 — введенная строка (локальная переменная);

k1 — общее количество цифр и знаков ‘+’, ‘*’, ‘–‘ в заданной строке s1 (фактический параметр).

Алгоритм решения задачи:

1) создаем функцию count, подсчитывающую общее количество цифр и знаков ‘+’, ‘*’, ‘–‘ в заданной строке;
2) вводим строку s1;
3) вызываем функцию count и выводим значение счетчика k1 на экран;

4) завершаем работу программы.

```
type str=string[100];
var s1 : str;
    k1 : integer;
function count(var s:str) : integer;
var i, k : integer;
begin
     k:=0;
     for i:=1 to length(s) do if s[i] in ['0'..'9', '+', '–', '*'] then k:=k+1;
     count:=k
end;
begin
    writeln('Введите строку');
    readln(s1);
    k1:=count(s1);
    writeln('Количество цифр и знаков – ', k1:5);
end.
```

Задача 2.9. *type M=set of 0..99;*

Описать функцию card(A), подсчитывающую количество элементов множества A типа M (ноль используйте как прекращение ввода). (Например, card([5,8,23])=3.)

Переменные:

а) в функции card:

A — множество (формальный параметр);

p — счетчик цикла от 0 до 99 (локальная переменная);

k — количество элементов множества A (локальная переменная);

б) в основной программе:

x — введенное число (локальная переменная);

b — созданное множество (локальная переменная);

k1 — количество элементов в созданном множестве (фактический параметр).

Алгоритм решения задачи:

1) создаем функцию card, подсчитывающую количество элементов множества;

2) организуем пустое множество;

3) в цикле repeat … until вводим числа до тех пор, пока не ввели 0, и дописываем их во множество;

4) вызываем функцию card и выводим значение счетчика k1 на экран;

5) завершаем работу программы.

```
type M=set of 0..99;
var b : m;
   k1, x : integer;
```

```
function card(A:M) : integer;
var p,k : integer;
begin
      k:=0;       for p:=0 to 99 do if p in A then k:=k+1;
      card:=k
end;
begin    b:=[ ];    repeat    write('Введите число >0<=99 – ');    readln(x);
b:=b+[x];    until x=0;    k1:=card(b);    writeln('Количество различных эле-
ментов множества', k1);
end.
```

Задача 2.10. *Программа. Дана строка из строчных латинских букв. Напечатать первые вхождения букв в текст, сохраняя их исходный взаимный порядок.*

Переменные:
let — множество малых латинских букв;
c — очередной символ строки;
s — введенная строка;
i — переменная цикла.

Алгоритм решения задачи:

1) организуем пустое множество let;
2) вводим строчку s;
3) организуем цикл, в котором просматриваем символы строки s до тех пор, пока не встретим ‘.’, и проверяем, входит ли этот символ во множество let (т.е. символ — малая латинская буква);
4) если очередной символ строки не входит во множество let, то этот символ встретился впервые; выводим его на экран и дописываем во множество;
5) завершаем работу программы.

```
var
      let : set of 'a'..'z';
      s : string;  c : char;  i : integer;
begin
      let:=[ ];      {множество букв в рассмотренной части текста}
      readln(s);
      i:=1;
      while s[i]<>'.' do
      begin if not(s[i] in let) then        {1-е вхождение}
                                  begin write(s[i]); c:=s[i]; let:=let+[c] end;
            i:=i+1;
      end;
      writeln
```

end.

Варианты заданий лабораторной работы № 2

Вариант 1	Задача 1 Даны целые числа a_1, a_2, ..., a_n. Все члены последовательности с четными номерами, предшествующие первому по порядку члену со значением max (a_1, a_2, ..., a_n), домножить на max (a_1, a_2, ..., a_n). Задача 2 Спортлото: 6 из 49. Составьте программу, в которой загадываются, иначе говоря, создаются 6 разных чисел, значения которых никак не связаны друг с другом, а величина лежит в интервале от 1 до 49. Указание. Для решения задачи используйте множества. Выражение random(49) +1 дает случайное целое число в интервале от 1 до 49.
Вариант 2	Задача 1 var k : integer; c : array[1..n, 1..m] of char; Определить k — количество различных элементов массива c (т.е. повторяющиеся элементы считать один раз). Указание: для хранения счетчиков для всех символов использовать массив типа array[char] of integer. Задача 2 В возрастающем порядке напечатать все целые числа из диапазона 1..255, представимые в виде $n^2 + m^2$, где m, $n \geq 0$. Указание. Для решения задачи используйте множества.
Вариант 3	Задача 1 var k : integer; c : array[1..n, 1..m] of integer; Определить k — количество «особых» элементов массива c, считая элемент «особым», если в его строке слева от него находятся элементы, меньшие его, а справа — большие. Указание: определите булевскую функцию f(x), которая проверяет, является ли элемент x «особым». Задача 2 Дана строка. В алфавитном порядке напечатайте (по разу) все малые латинские буквы, входящие в эту строку ровно один раз. Указание: решение задачи простое, если вы будете использо-

	вать множества.
Вариант 4	Задача 1 Дана матрица целых чисел размером MxN. Найти номера строки и столбца наибольшего элемента матрицы. Задача 2 Дана строка. В алфавитном порядке напечатайте все малые латинские буквы, не входящие в эту строку. Указание: решение задачи простое, если вы будете использовать множества.
Вариант 5	Задача 1 Даны действительные числа a_1, a_2, ..., a_n, b_1, b_2, ..., b_n. Вычислить $(a_1 + b_n) * (a_2 + b_{n-1})* ... *(a_n + b_1)$. Задача 2 Спортлото: 5 из 36. Составьте программу, в которой загадываются, иначе говоря, создаются 5 разных чисел, значения которых никак не связаны друг с другом, а величина лежит в интервале от 1 до 36. Указание: решение задачи простое, если вы будете использовать множества. Выражение random(36) +1 дает случайное целое число в интервале от 1 до 36.
Вариант 6	Задача 1 Даны координаты n точек на плоскости: x_1, y_1, ..., x_n, y_n. Найти номера двух точек, расстояние между которыми наибольшее (считать, что такая пара точек единственная). Задача 2 Напечатать все натуральные числа от 10 до 32767, в десятичной записи которых нет одинаковых цифр. Указание: решение задачи простое, если вы будете использовать множества. Необходимо для каждого числа создавать множество, состоящее из цифр числа. При этом проверять: если очередная цифра числа есть уже во множестве, то такое число не надо выводить на экран.
Вариант 7	Задача 1 Определить, является ли заданная целая квадратная матрица n-го порядка магическим квадратом, т.е. такой, в которой суммы элементов во всех строках и столбцах одинаковы. Пример магического квадрата: 0 1 2 3 4 1 2 3 4 0 2 3 4 0 1 3 4 0 1 2

	4 0 1 2 3 Задача 2 Дана строка. В алфавитном порядке напечатайте (по разу) все малые латинские буквы, входящие в эту строку более одного раза. Указание: решение задачи простое, если вы будете использовать множества.
Вариант 8	Задача 1 Дана последовательность из n целых чисел. Определить количество инверсий в этой последовательности (т.е. таких пар элементов, в которых большое число находится слева от меньшего: $x_i > x_j$ при $i < j$). Задача 2 Дана строка. В алфавитном порядке напечатайте (по разу) все строчные латинские согласные буквы, входящие в эту строку. Указание: гласные буквы — a, e, i, o, u; остальные — согласные. Решение задачи простое, если вы будете использовать множества.
Вариант 9	Задача 1 type вектор = array [1..n] of integer; (n – четно) матрица = array [1..n] of вектор; var A : матрица; x : вектор; В матрице A поменять местами 1-ю и 2-ю строки, 3-ю и 4-ю строки,..., (n–1)-ю и n-ю строки (воспользоваться x как вспомогательным массивом). Задача 2 Дана строка символов. Подсчитайте количество различных латинских малых букв, входящих в данную строку. Указание: решение задачи простое, если вы будете использовать множества.
Вариант 10	Задача 1 var A : array [1..6,1..6] of boolean; B : array [1..5,1..5] of boolean; n,k : 1..6; Получить массив B из массива A удалением n-ой строки и k-го столбца. Задача 2 Дана строка. В алфавитном порядке напечатайте (по разу) все большие латинские гласные буквы, входящие в эту стро-

	ку. Указание: гласные буквы — a, e, i, o, u; остальные — согласные. Решение задачи простое, если вы будете использовать множества.
Вариант 11	Задача 1 const n =..; var A,B,C : array [1..n,1..n] of real; Вычислить C = $(A+B)^2$ (сумма двух матриц A и B возводится в квадрат). Указание: определите процедуру с параметрами для сложения матриц и процедуру с параметрами для возведения матрицы в квадрат. Задача 2 Дана строка из строчных латинских букв. Напечатайте все буквы, входящие в текст не менее двух раз. Указание. Просматривая в цикле символы текста, формируйте два множества: одно содержит уже просмотренные символы, другое наполняется теми элементами, которые входят в первое множество.
Вариант 12	Задача 1 const n = ...; var x, y : array [1..n] of real; Дано k (1 <= k <= n–1). Преобразовать массив x по следующему правилу (воспользоваться массивом y как вспомогательным): элементы массива x циклически сдвинуть на k позиций влево. Подсказка: определить процедуру для сдвига массива на 1 позицию влево и применить ее k раз. Задача 2 Дана строка. Определить, каких букв больше в этом тексте: латинских строчных гласных или согласных. Указание: гласные буквы — a, e, i, o, u; остальные — согласные. Решение задачи простое, если вы будете использовать множества.
Вариант 13	Задача 1 var A : array [1..n,1..n] of real; Найти сумму элементов из области матрицы A, отмеченной символом '*' (диагонали входят в выделенную область): * * * * * * * 0 * * * * * 0 0 0 * * * 0 0

	0 0 0 * 0 0 0 0 0 * * * 0 0 0 * * * * * 0 * * * * * * * Задача 2 Дана строка. В алфавитном порядке напечатайте (по разу) все малые латинские буквы, входящие в эту строку ровно два раза. Указание: решение задачи простое, если вы будете использовать множества. Заведите три множества X1, X2 и X3 (сначала эти множества пустые). И в цикле просмотрите все символы из введенной строки. С каждым символом — маленькой латинской буквой — выполняйте следующие действия: 1. Если этого символа нет в X1, то поместите его туда (после окончания цикла множество X1 содержит все символы из строки). 2. Если этот символ есть в X1 и нет в X2, то поместите этот символ в X2 (после окончания цикла множество X2 содержит все символы из строки, которые там встречаются более одного раза). 3. Если этот символ есть в X1 и в X2, то поместите этот символ в X3 (после окончания цикла множество X3 содержит все символы из строки, которые там встречаются более двух раз). После окончания цикла распечатайте в алфавитном порядке все элементы из разности множеств X2–X3. Это будет ответ. (Используйте цикл for c:=’a’ to ‘z’ do если c входит в X2–X3, то печать c.)
Вариант 14	Задача 1 Даны натуральное n и (построчно) элементы квадратной вещественной матрицы A четвертого порядка. Вычислить n-ю степень этой матрицы ($A^1 = A$; $A^2 = A*A$; $A^3 = A^2 * A$ и т.д.). Указания: 1. Определите процедуру mult(a,b,c) для умножения матриц (a*b = c). 2. Для вычисления n-ой степени матрицы поступайте так же, как при вычислении в цикле n-ой степени числа, но вместо умножения чисел используйте процедуру mult.

	Задача 2 Дана строка. В алфавитном порядке напечатайте (по разу) все малые латинские гласные буквы, входящие в эту строку ровно 3 раза. Указание: гласные буквы — a, e, i, o, u; остальные — согласные. Решение задачи простое, если вы будете использовать множества.
Вариант 15	Задача 1 Даны действительные числа a_1, a_2, ..., a_n, a_n , ... , a_{2n} . Получить max ($a_1 + a_{2n}$, $a_2 + a_{2n-1}$, ... , $a_n + a_{n+1}$); min($a_1 * a_n$, $a_2 * a_{n+1}$, ..., $a_{n+1} * a_{2n}$). Задача 2 Напечатать все натуральные числа от 10 до 32767, в десятичной записи которых нет одинаковых цифр. Указание: решение задачи простое, если вы будете использовать множества. Необходимо для каждого числа создавать множество, состоящее из цифр числа. При этом проверять: если очередная цифра числа есть уже во множестве, то такое число не надо выводить на экран.
Вариант 16	Задача 1 const n = ...; var s : array [1..n] of integer; Напечатать те элементы массива s, которые являются полными квадратами (1, 4, 9, 16, 25,...). Подсказка. Число k является полным квадратом, если выполнено равенство $k = (\text{trunc}(\sqrt{k}))^2$. Задача 2 Дана строка из строчных латинских букв. Напечатать первые вхождения букв в текст, сохраняя их взаимный порядок. Указание: формируйте множество, в которое очередной элемент добавляется после первой печати.
Вариант 17	Задача 1 Даны целые числа a_1, a_2, ..., a_n, каждое из которых отлично от нуля. Если в последовательности отрицательные и положительные члены чередуются (+, –, +, –,.... или –, +, –, +,...), то напечатать исходную последовательность. Иначе напечатать все отрицательные члены последовательности, сохранив порядок их следования. Задача 2 Введите множество, состоящее из 20 целых чисел от 1 до 50. Определите, сколько чисел, у которых последняя цифра 3, 5 или 7.

Вариант 18	Задача 1 Напечатать величины a_1, a_2, ..., a_n, где a_0 — заданное целое число, $a_i = a_{[i/2]} + a_{i-1}$ при i = 1, 2, ... n. [i/2] означает целую часть числа i/2. Задача 2 type S = set of 0..50; Описать функцию p (A), подсчитывающую произведение элементов множества A типа S. (Например, sum ([5,8,23] = 920.)) Программа: введите множество A (ноль используйте как прекращение ввода). Выдайте, вызвав функцию p, произведение элементов множества A.
Вариант 19	Задача 1 var A : array [1..n,1..n] of real; (n–нечетно) Найти сумму элементов из области матрицы A, отмеченной символом ' * ': 1 n 0 0 0 * 0 0 0 0 0 * * * 0 0 0 * * * * * 0 * * * * * * * 0 * * * * * 0 0 0 * * * 0 0 n 0 0 0 * 0 0 0 Задача 2 Создайте множество, состоящее из **простых** чисел < 255 и дающее при делении на 4 остаток 3. Напечатайте элементы этого множества.
Вариант 20	Задача 1 Даны целые числа a_1, a_2, ..., a_n. Наименьший член последовательности a_1, a_2, ..., a_n заменить целой частью среднего арифметического всех членов, остальные члены оставить без изменения. Если в последовательности несколько членов со значением min(a_1, a_2, ..., a_n), то заменить последний по порядку. Задача 2 Дана строка символов. В возрастающем порядке напечатайте все цифры, входящие в эту строку.

ЛАБОРАТОРНАЯ РАБОТА № 3

Лабораторная работа № 3 состоит из двух задач. Первая задача посвящена созданию программ, отражающих приемы работы с файлами. Необходимо в начале программы создать файл (элементы файла вводятся с клавиатуры или генерируются случайным образом), а в конце программы сначала вывести на экран содержимое созданного файла, а затем отразить полученные результаты. Во второй задаче необходимо составить программу, реализующую рекурсивный алгоритм.

В ходе выполнения лабораторной работы № 3 необходимо составить программы на языке Паскаль.

Примеры решения задач

Файлы

Задача 3.1. *type series=file of real;Описать функцию neg(s), подсчитывающую сумму отрицательных элементов в файле s типа series.*

Переменные:
а) основная программа:
f — файл вещественных чисел (глобальная переменная);
y — очередное число для записи в файл (глобальная переменная);
n — сумма отрицательных элементов файла (глобальная переменная);
k — счетчик цифр.
б) функция neg:
s — файловая переменная (локальная, формальная переменная);
x — очередной элемент файла (локальная переменная);
sum — сумма отрицательных элементов файла (локальная переменная).

Алгоритм решения задачи:

1) свяжем файл proba.txt с файловой переменной f и откроем его для записи;
2) организуем цикл, в котором вводим числа и записываем их в файл до тех пор, пока не введем 0;
3) вызываем функцию neg, в которой открываем файл для чтения, считываем последовательно элементы файла и определяем сумму отрицательных элементов файла;
4) выводим значение суммы на экран.

```
type series=file of real;
var f : series;
    n,y : real;
```

```
function neg(var s : series) : real;
var sum,x : real;
begin
     reset(s); sum:=0;
     while not eof(s) do
     begin
          read(s,x);
          if x<0 then sum:=sum+x
     end;
     neg:=sum ; close(s)
end;
begin
    assign (f,'proba.txt');
    rewrite(f);
    repeat
          write('Введите число (0 – конец ввода)– ');
          readln(y);
          write(f,y);
    until (y=0);
    close (f);
    n:=neg(f);
    writeln('Сумма отрицательных элементов файла n= ',n:10:3);
            close(f);
end.
```

Задача 3.2. *type row=file of 0..999;*

Описать логическую функцию sort(r), проверяющую, упорядочены ли по возрастанию элементы непустого файла r типа row.

```
type row1 = 0..999;
      row = file of row1;

var r : row;
   x,y : row1;
   i,k : integer;
function sort(var r : row) : boolean;
var x,y : row1; ok : boolean;
begin
     reset(r);
     read(r,y); ok:=true;
     while not eof(r) and ok do
          begin x:=y; read(r,y);  ok:=x<y end;
     sort:=ok; close(r)
end;
```

```
begin
    write('Введите количество элементов файла');
    readln(k);
    assign(r,'12.dat');
    rewrite(r);
    for i:= 1 to k do
    begin
       writeln('Введите ',i,'-ый элемент файла');
       read(x);  write(r,x);
    end;
    close(r);
    if sort(r) then writeln('Упорядочены по возрастанию')
              else writeln('Не упорядочены по возрастанию');
end.
```

Задача 3.3. *type reals=file of real;*

Описать функцию less(f) от непустого файла f типа reals, которая подсчитывает количество элементов файла f, меньших среднего арифметического всех элементов этого файла.

```
type reals = file of real;
var f1 : reals;
    a : real;l,i : integer;
function less(var f : reals) : integer;
var k : integer; x,s : real;
begin                          {подсчет среднего арифметического:}
    reset(f); k:=0; s:=0;
    repeat
        read(f,x);
        k:=k+1;
        s:=s+x
    until eof(f);
    s:=s/k;
   {новый просмотр f и подсчет элементов <s:}
    reset(f); k:=0;
    repeat
         read(f,x);
         if x<s then k:=k+1
    until eof(f);
    less:=k; close(f)
end;
begin
    assign(f1,'c:\tp\work\23.dat');
```

```
   rewrite(f1);
   for i:=1 to 10 do
       begin readln(a);  write(f1,a); end;
    l:=less(f1); close(f1);
    writeln('l=', l);
end.
```

Рекурсия

Задача 3.4. *Использовать рекурсию для нахождения цифрового корня целого числа.*

Цифровой корень находится суммой через сумму цифр числа до тех пор, пока эта сумма сама не станет цифрой. Например, для числа 9999999 цифровой корень находится так:

9+9+9+9+9+9+9 = 63;

6+3 = 9.

Цифровой корень 9999999 равен девяти.

Переменные:

а) в функции num:

n — целое число (глобальная переменная);

s — вспомогательная переменная (локальная переменная);

б) в функции root:

n — целое число (глобальная переменная);

в) в основной программе:

n — целое число (глобальная переменная).

Алгоритм решения задачи:

1) создаем описание функций num и root;

2) вводим целое число n;

3) вызываем рекурсивную функцию root и определяем цифровой корень числа n;

4) завершаем работу программы.

```
var n:longint;
function num(i:longint):integer;
 var  s: integer;
begin
  s:=0;
  repeat
    s:=s+n mod 10;
    n:= n div 10;
 until n=0;
```

```
 num:=s;
end;
function root(l:longint):integer;
begin
   if n<10 then root:=n
      else
        begin
        n:=num(n);
        root:=root(n);
        end;
end;

begin
  write('Введите целое число n=');
  readln(n);
  writeln('Его цифровой корень равен : ', root(n));
end.
```

Задача 3.5. *Напишите рекурсивную программу вычисления суммы* $\sum_{i=2}^{n} i^2 + i + 5$

```
var n : integer;
function sum(i : integer) : real;
begin
    if i=1 then sum:=0 else sum:=sum(i–1)+i*i+i+5;
end;

begin
writeln('Введите n');
readln(n);
writeln('Значение суммы равно – ', sum(n));
end.
```

Варианты заданий лабраторной работы № 3

Вариант 1	Задача 1 type ряд = file of integer; Описать процедуру p(f,g) от двух файлов типа ряд, которая в пустой файл f переписывает положительные элементы файла g. Указание: введите файл g (последний элемент равен 0). Выполните процедуру p(f,g). Распечатайте файл f. Задача 2 Напишите рекурсивную подпрограмму, которая печатает в обратном порядке заданную строку.

Вариант 2	Задача 1
	type fr = file of real;
	Описать процедуру predlast(f), значением которой является предпоследний элемент файла f, имеющего тип fr и содержащего не менее двух элементов.
	Указание: введите файл f (последний элемент = 0) и вычислите функцию predlast(f).
	
	Задача 2
	type
	reals = file of real;
	var f : reals;
	Опишите рекурсивную функцию sum без параметров для нахождения суммы элементов файла f.

Вариант 3	Задача 1 type rad = file of integer; Описать процедуру p(f,g) от двух файлов типа rad, которая из файла f переписывает в пустой файл g сначала все положительные числа, а потом все отрицательные. Указание: введите файл f (последний элемент равен 0). Выполните процедуру p(f,g). Распечатайте файл g. Задача 2 Описать рекурсивную функцию letter (s), которая подсчитывает количество букв в строке s.
Вариант 4	Задача 1 type seria = file of integer; Описать функцию prod(s) с вещественным значением, подсчитывающую произведение ненулевых элементов файла s. Указание: введите файл s (последний элемент равен 0) и напечатайте значение функции prod(s).

	Задача 2 Напишите рекурсивную программу, которая n раз выводит на экран текст истории о попе и его собаке (рассказ в рассказе).
Вариант 5	Задача 1 type seria = file of integer; var s:seria; Описать функцию max(s), вычисляющую значение максимального элемента файла s. Указание: введите файл s (последний элемент равен 0) и напечатайте значение функции max(s). Задача 2 Во входном файле задана непустая последовательность положительных вещественных чисел, за которой следует отрицательное число. Описать рекурсивную функцию sum без параметров для нахождения суммы этих положительных чисел.
Вариант 6	Задача 1 Описать логическую функцию check(s) для текстового файла s, проверяющую, содержит ли файл s латинские буквы. Указание: введите файл s (последний элемент в файле '.') и вычислите значение функции check(s). Задача 2 type reals = file of real; var f : reals; Напишите рекурсивную программу, которая печатает сначала все отрицательные элементы этого файла, а затем положительные (в любом порядке).
Вариант 7	Задача 1 Для файла s, состоящего из целых чисел, определите процедуру, которая находит значения максимального и минимального элементов файла s. Указание: введите файл s (последний элемент равен 0) и напечатайте значения максимального и минимального элементов файла s. Задача 2 const n=...; type vector = array [1..n] of real; Описать функцию min (x) для определения минимального

	элемента вектора x, введя вспомогательную рекурсивную функцию min1 (k), находящую минимум среди последних элементов вектора x, начиная с k-го. Указание: пусть min(x) — функция, которая вычисляет минимум среди элементов массива x, а min1(k) — функция, которая вычисляет минимум среди элементов x_k, x_{k+1}, …, x_n массива x (для этой функции массив x есть глобальная переменная, определенная в главной программе). Имеем равенство min(x) = min1(1). Определим функцию min1(k) рекурсивно: 1) если k=n, то функция возвращает результат x_n; 2) если k<n, то функция возвращает наименьшее из двух чисел x_k и min1(k+1) (второе число получается при рекурсивном вызове).
Вариант 8	Задача 1 type rad = file of 1..maxint; Описать процедуру prim(f,n), записывающую в файл f все простые числа 2,3,5,7,11,13,17..., не превосходящие целого положительного числа n. Указание: введите n. Выполните процедуру prim(f,n). Распечатайте файл f. Задача 2 Описать рекурсивную функцию sign (s), которая подсчитывает количество знаков препинания в строке s (знаки: '.' , ',' , ';' , ':').
Вариант 9	Задача 1 type ряд = file of integer; Описать процедуру append(f,g,h) от трех файлов типа ряд, которая записывает в файл f сначала все элементы файла g, а затем все элементы файла h. Указание: введите два файла g и h. Выполните процедуру append(f,g,h). Напечатайте файл f. Задача 2 Дан массив a : array [1..n] of integer; Напишите рекурсивную программу для вычисления суммы $$\sum_{i=1}^{n} 1/a[i]$$ Указание: Пусть функция f(k) вычисляет сумму 1/a[1]+1/a[2]+…+1/a[k].

	Нам надо вычислить f(n). Определим f(k) рекурсивно: 1) если k=1, то f(k)=1/a[1]; 2) если k>1, то значение функции равно сумме 1/a[k] и f(k–1).
Вариант 10	Задача 1 Type fr = file of real; Описать функцию s3(f), значением которой является сумма последних трех элементов файла f, имеющего тип fr и содержащего не менее 3 элементов. Указание: введите файл f (последний элемент = 0) и вычислите функцию s3(f). Задача 2 type reals = file of real; var f : reals; Опишите рекурсивную функцию sum (n) для нахождения суммы $\sum_{i=1}(a[i])^n$, где a[i] — элемент файла f. Указание. Пусть функция f(k) вычисляет сумму $a[1]^n+a[2]^n+\ldots+a[k]^n$. Нам надо вычислить f(n). Определим f(k) рекурсивно: 1) если k=1, то $f(k)=a[1]^n$; 2) если k>1, то значение функции равно сумме $a[k]^n$ и f(k–1).
Вариант 11	Задача 1 type seria = file of integer; var s:seria; Описать функцию poz(s), подсчитывающую количество положительных чисел в файле s. Указание: введите файл s (последний элемент равен 0) и напечатайте значение функции poz(s). Задача 2 Напишите рекурсивную программу для вычисления суммы $\sum_{i=1}^{n}1/(i)^2$. Указания. Пусть функция f(k) вычисляет сумму a[1]+a[2]+…a[k]. Нам надо вычислить f(n). Определим f(k) рекурсивно:

	1) если k=1, то f(k)=a[1]; 2) если k>1, то значение функции равно сумме a[k] и f(k–1).
Вариант 12	Задача 1 Описать процедуру letter(s,t), которая записывает в текстовый файл t все латинские буквы из строки s. Указание: введите строку s. Выполните процедуру letter(s,t) и распечатайте элементы файла t. Задача 2 Функция f (n) определена для целых положительных чисел следующим образом: $$f(n) = \begin{cases} 1, & \text{åñëè } n = 1; \\ \sum_{i=2}^{n} f(n \text{ div } i), & \text{åñëè } n \geq 2. \end{cases}$$ Вычислить f (k) для k=15, 16,..., 30.
Вариант 13	Задача 1 type reals = file of real; Описать процедуру p(f,g,h) от трех файлов типа reals, которая переписывает из файла f в непустой файл g все элементы, меньшие среднего арифметического всех элементов файла f, и в непустой файл h записывает все остальные числа. Указание: введите файл f (последний элемент = 0). Выполните процедуру p(f,g,h). Распечатайте файлы g и h. Задача 2 Напишите рекурсивную программу для вычисления суммы $$\sum_{i=2}^{n} \frac{1}{(i+1) * i * (i-1)}.$$ Указание: Пусть функция f(k) вычисляет сумму a[1]+a[2]+…a[k]. Нам надо вычислить f(n). Определим f(k) рекурсивно: 1) если k=1, то f(k)=a[1]; 2) если k>1, то значение функции равно сумме a[k] и f(k–1).
Вариант 14	Задача 1 Дан текстовый файл, в котором строки содержат как латинские буквы, так и цифры. Необходимо создать другой текстовый файл, содержащий строки из первого, преобразованные по следующему принципу: в начале строки расположены все буквы исходной строки, а затем все цифры (в том же порядке). Описать процедуру letter(s,t), которая записывает в тексто-

	вый файл t сначала все латинские буквы из строки s, а затем все цифры. Указание: введите исходный файл. Выполните процедуру letter(s,t) и распечатайте элементы файла t. Задача 2 Дан массив a : array [1..n] of integer; Напишите рекурсивную программу для вычисления произведения $$\prod_{i=1}^{n} a[i].$$ Указание. Пусть функция f(k) вычисляет произведение a[1]*a[2]*…a[k]. Нам надо вычислить f(n). Определим f(k) рекурсивно: 3) если k=1, то f(k)=a[1]; 4) если k>1, то значение функции равно произведению a[k] и f(k–1).
Вариант 15	Задача 1 type seria = file of integer; var s:seria; Описать функцию poz(s, n), подсчитывающую количество положительных и отрицательных чисел в файле s. Указание: введите файл s (последний элемент равен 0) и напечатайте значение функции poz(s, n). Задача 2 Описать рекурсивную функцию digits (s), которая подсчитывает сумму цифр в строке s.

ЛАБОРАТОРНАЯ РАБОТА № 4

Лабораторная работа № 4 состоит из двух задач. Первая задача посвящена созданию программ, отражающих приемы работы со списками. Второе задание посвящено созданию программ в графическом режиме.

В ходе выполнения лабораторной работы № 4 требуется составить программы на языке Паскаль.

Примеры решения задач

Списки

Задача 4.1. *type telem='a'..'z'.*
list=^node;
node= record
info:telem;
next:list
end;

Пусть E1 и E2 — данные типа telem.
Описать функцию или процедуру, которая :
а) заменяет в списке L все вхождения E1 и E2;
б) проверяет, упорядочены ли элементы списка L по алфавиту

а)

```
type telem='a'..'z';
        list=^node;
        node= record
              info : telem;
              next : list
             end;
var s,l : list;
   x,e,e1 : telem;
   n,i : integer;

procedure change (l : list; e,e1: telem);
var p:list;                              {ссылка на очередное звено}
begin
     p:=L;
     while p<>nil do
     begin
          if p^.info=e then p^.info:=e1;
          p:=p^.next                     {переход к следующему звену}
    end;
end;
```

```
procedure out_spisok(l : list);              {выводит список на экран}
begin
    while l<> nil do
    begin
        s:=l^.next;
       write(l^.info,' ');
       l:=s;
    end;
    writeln;
end;

begin
                                                    {формируем список}
    s:=nil;
    writeln('Введите количесто элементов списка');
    readln(n);
    for i:=1 to n do
    begin
        new(l);
        l^.next:=s;
        readln(x);
        l^.info:=x;
        s:=l;
    end;
                                                     {выводим список на экран}
    writeln('Введенный список');
    out_spisok(l);
                                                {заменяем элемент E на E1}
    writeln('Введите элемент, который Вы хотите заменить');
    readln(e);
    writeln('Введите элемент, на который Вы хотите заменить');
    readln(e1);
    change (l,e,e1);
    writeln('Полученный список');
    out_spisok(l);
                                                {освобождаем динамическую память}
    while l<> nil do
    begin
        s:=l^.next;
        dispose(l);
        l:=s;
    end;
end.
```

б)

```
type telem='a'..'z';
        list=^node;
        node= record
              info : telem;
              next : list
             end;
var s,l : list;
   x : telem;
   n,i : integer;

function sort(l : list) : boolean;
var p,q : list;                                  {ссылка на пару соседних звеньев}
     ok : boolean;
begin
     ok:=true;
     p:=L;                                       {nil или ссылка на 1-звено}
     if p<>nil then
                 begin q:=p^.next;               {nil или ссылка на 2-е звено}
                       while (q<>nil) and ok do
                       begin
                            ok:=p^.info<=q^.info;
                            p:=q; q:=q^.next     {переход к след. паре}
                       end
                 end;
     sort:=ok
end;

procedure out_spisok(l:list);
begin
  while l<> nil do
  begin
  s:=l^.next;
  write(l^.info,' ');
  l:=s;
  end;
  writeln;
end;

begin
                                                 {формируем список}
  s:=nil;
  writeln('Введите количесто элементов списка');
```

```
  readln(n);
  for i:=1 to n do
  begin
  new(l);
  l^.next:=s;
  readln(x);
  l^.info:=x;
  s:=l;
  end;
                                                    {выводим список на экран}
  writeln('Введенный список');
  out_spisok(l);
  if sort(l) then writeln('Список отсортирован по алфавиту')
          else writeln('Список не отсортирован по алфавиту');
                                                  {освобождаем динамическую память}
  while l<> nil do
  begin
  s:=l^.next;
  dispose(l);
  l:=s;
  end;
end.
```

Задача 4.2. *type telem=...;*
list=^node;
node=record
info : telem;
next : list
end;

Описать процедуру, которая вставляет в список L новый элемент E1 перед первым вхождением элемента E, если E входит в L;

```
type telem=0..999;
        list=^node;
        node= record
              info : telem;
              next : list
             end;
var s,l : list;
   x,e,e1 : telem;
   n,i : integer;

procedure insert( l : list; e,e1 : telem);
```

```
  var p,q : list; eq : boolean;
begin                                                    { поиск звена с E:}
    p:=L; eq:=false;
    while (p<>nil) and not eq do
       if p^.info=e then eq:=true
                    else p:=p^.next;
     if eq then                              { вставка E1 перед E}
            begin                              {внимание-трюк: запись E1 в звено p
                                                вместо E и вставка E за звеном p:}
              p^.info:=e1; new(q); q^.info:=e;
              q^.next:=p^.next; p^.next:=q
            end;
end;

procedure out_spisok(l : list);
begin
  while l<> nil do
  begin
  s:=l^.next;
  write(l^.info,' ');
  l:=s;
  end;
  writeln;
end;

begin
                                                         {формируем список}
  s:=nil;
  writeln('Введите количесто элементов списка');
  readln(n);
  for i:=1 to n do
  begin
  new(l);
  l^.next:=s;
  readln(x);
  l^.info:=x;
  s:=l;
  end;
                                                       {выводим список на экран}
  writeln('Введенный список');
  out_spisok(l);
  writeln('Введите элемент, перед которым Вы хотите вставить число');
  readln(e);
```

```
  writeln('Введите элемент, который Вы хотите вставить в список');
  readln(e1);
  insert(l,e,e1);
  writeln('Полученный список');
  out_spisok(l);
                                          {освобождаем динамическую память}
  while l<> nil do
  begin
  s:=l^.next;
  dispose(l);
  l:=s;
  end;
end.
```

Задача 4.3. *Описать процедуру, которая удаляет из непустого списка l последний элемент.*

```
type list=^node;
    node= record
          info : integer;
          next : list
          end;
var s,l : list;
   x : integer;
   n,i : integer;

procedure del(var l : list);
var p,q : list;
begin
     if l=nil then {удалять нечего}
               else  if l^.next=nil {в списке один элемент}
               then
                     begin dispose(l);l:=nil end
                      else begin
                            {поиск предпослед.(p) и послед.(q) звеньев: }
                             p:=l; q:=p^.next;
                             while q^.next<>nil do
                              begin p:=q; q:=q^.next end;
                             {удаление последнего звена:}
                              dispose(q); p^.next:=nil
                     end;
end;
```

```
procedure out_spisok(l : list);
begin
  while l<> nil do
  begin
  s:=l^.next;
  write(l^.info,' ');
  l:=s;
  end;
  writeln;
end;

begin
                                                    {формируем список}
  s:=nil;
  writeln('Введите количесто элементов списка');
  readln(n);
  for i:=1 to n do
  begin
  new(l);
  l^.next:=s;
  readln(x);
  l^.info:=x;
  s:=l;
  end;
                                                    {выводим список на экран}
  writeln('Введенный список');
  out_spisok(l);
  del(l);
  writeln('Полученный список');
  out_spisok(l);
                                                    {освобождаем динамическую память}
  while l<> nil do
  begin
  s:=l^.next;
  dispose(l);
  l:=s;
  end;
end.
```

Задача 4.4. *Описать рекурсивную функцию или процедуру, которая:*
а) определяет, входит ли элемент E в список L;
б) удаляет из списка L первое вхождение элемента E, если такое есть;

a)

```
type list=^node;
     node= record
           info : integer;
           next : list
           end;
var s,l : list;
    x,e : integer;
    n,i : integer;

function memb(l : list; e : integer) : boolean;
var l1 : list;
begin
   if l=nil then memb:=false
            else   if l1^.info=e then memb:=true
                                 else memb:=memb(l1^.next,e)
end;

procedure out_spisok(l : list);
begin
   while l<> nil do
   begin
   s:=l^.next;
   write(l^.info,' ');
   l:=s;
   end;
   writeln;
end;

begin
                                                    {формируем список}
   s:=nil;
   writeln('Введите количесто элементов списка');
   readln(n);
   for i:=1 to n do
   begin
   new(l);
   l^.next:=s;
   readln(x);
   l^.info:=x;
   s:=l;
   end;
                                                  {выводим список на экран}
```

```
  writeln('Введенный список');
  out_spisok(l);
  writeln('Введите интересующий Вас элемент');
  readln(e);
  if not(memb(l,e)) then writeln('Элемент ', e, ' входит в список')
                    else writeln('Элемент ', e, ' не входит в список');
                                        {освобождаем динамическую память}
  while l<> nil do
  begin
  s:=l^.next;
  dispose(l);
  l:=s;
  end;
end.
```

б)

```
type list=^node;
    node= record
          info : integer;
          next : list
          end;
var s,l : list;
   x,e : integer;
   n,i : integer;

procedure delete(var l : list; e :integer);
var p : list;
begin
    if l<>nil then
                begin
                if l^.info=e then                          {удалить первое звено}
                            begin p:=l; l:=L^.next; dispose(p) end
                            else          {удалить E из "хвоста" списка и записать
                                    в 1-е звено ссылку на измененный  хвост":}
                            delete(l^.next,e)
              end;
end;


procedure out_spisok(l : list);
begin
  while l<> nil do
  begin
```

```
  s:=l^.next;
  write(l^.info,' ');
  l:=s;
  end;
  writeln;
end;

begin
   {формируем список}
  s:=nil;
  writeln('Введите количесто элементов списка');
  readln(n);
  for i:=1 to n do
  begin
  new(l);
  l^.next:=s;
  readln(x);
  l^.info:=x;
  s:=l;
  end;
  {выводим список на экран}
  writeln('Введенный список');
  out_spisok(l);
  writeln('Введите интересующий Вас элемент');
  readln(e);
  delete(l,e);
  writeln('Полученный список');
  out_spisok(l);
 {освобождаем динамическую память}
  while l<> nil do
  begin
  s:=l^.next;
  dispose(l);
  l:=s;
  end;
end.
```

Программы, работающие в графическом режиме

Для решения задач, приведенных ниже, используется модуль GRAPH, этот модуль не может быть использован для написания программ в системе PascalABC, так как для данной системы разработан модуль GRAPHABC, процедуры и функции которого не всегда совпадают с про-

цедурами и функциями модуля GRAPH. В примерах 8 и 9 данного раздела приводятся программы, использующие модуль GRAPHABC, также примеры использования данного модуля можно посмотреть в разделе **Помощь** системы программирования PascalABC.

Задача 4.5. *Построить семейство одинаковых окружностей, центры которых лежат на вертикально вращающемся отрезке, верхний конец которого закреплен.*

Переменные:
x, y — координаты центра очередного маленького круга;
y0 — смещение кругов по вертикале;
i — переменная цикла;
t — угол поворота;
drive — тип графического драйвера;
mode — режим работы графического адаптера.

Для решения задачи:
1) инициируем модуль graph;
2) устанавливаем начальные значения радиуса, координаты центра;
3) организуем цикл, в котором закрашиваем круги со все большим радиусом до тех пор, пока не будет нажата любая клавиша.

```
uses crt, graph;
var drive,mode,x,y,i,t,y0:integer;
begin
  drive:=detect;
  initgraph(drive,mode,'c:\tp\bgi');
  setfillstyle(1,1);
  floodfill(1,4,1);
  t:=–4;
  y0:=10;
  setcolor(16);
  for i:=1 to 150 do
  begin
     t:=t+2;
     y0:=y0+3;
     x:=getmaxx div 2 + trunc(cos(t/10)*i);
     y:=y0 – trunc(sin(t/10)*i);
     setfillstyle(1,14);
     fillellipse(x,y,20,20);
     delay(100);
  end;
```

```
  repeat
  until keypressed;
  closegraph;
end.
```

Задача 4.6. *Построить движущиеся изображения НЛО на фоне звездного неба.*

I. Переменные:

x, y — случайные координаты;

r — радиус;

i — переменная цикла;

drive — тип графического драйвера;

mode — режим работы графического адаптера.

Алгоритм решения задачи:

1) инициируем модуль graph;

2) организуем безусловный цикл по переменной i и рисуем звездное небо;

3) организуем цикл до тех пор, пока не будет нажата любая клавиша;

4) в этом цикле рисуем НЛО с помощью двух эллипсов, двух линий и двух маленьких кружочков, держим на экране, затем стираем изображение процедурой CLEARDEVICE;

5) опять рисуем звездное небо;

6) определяем случайным образом координаты следующего изображения НЛО;

7) после нажатия любой клавиши закрываем графический режим.

```
uses crt,graph;
var drive,mode,x,y,i,r:integer;
begin
  r:=40;
  x:=r*5;
  y:=r*2;
  drive:=detect;
  initgraph(drive,mode,'c:\tp\bgi');
  setcolor(3);
  for i:=1 to 600 do
    putpixel(random(i),random(i),i);
  repeat
     ellipse(x,y,0,360,r,(r div 3)+2);
     ellipse(x,y–4,190,357,r,r div 3);
     line(x–17,y–16,x–25,y–22);
     line(x+17,y–16,x+25,y–22);
```

```
      circle(x+25, y–25,2);
      circle(x–25, y–25,2);
      setfillstyle(1,3);
      floodfill(x+1,y+4,3);
      delay(150);
      cleardevice;
      for i:=1 to 600 do
         putpixel(random(i),random(i),i);
      x:=x+random(10);
      y:=y+random(10);
   until (keypressed);
  closegraph;
end.
```

II. Эта же программа может быть написана с использованием пары процедур GETIMAGE(lx,ly,rx,ry,saucer^) и PUTIMAGE(x,y,saucer^,xorput).

Процедура GETIMAGE(lx,ly,rx,ry,saucer^) помещает изображение в буфер, а PUTIMAGE(x,y,saucer^,xorput) выводит в заданное место изображение.

Параметр xorput определяет способ вывода на экран — исключающее ИЛИ.

Например, операторами

```
GETIMAGE(lx,ly,rx,ry,saucer^);
READLN;
PUTIMAGE(x,y,saucer^,xorput);
```

мы выводим изображение на экран и после нажатия любой клавиши стираем его.

Можно использовать другие способы вывода изображения на экран, например:

NORMALPUT — стирает часть экрана и на это место выводит изображение;

NOTPUT — делает то же самое, но изображение инвертируется;

ORPUY — дописывает новое изображение.

Переменные:

x, y — случайные координаты;

r — радиус;

i — переменная цикла;

saucer — указатель буфера хранения изображения;

drive — тип графического драйвера;

mode — режим работы графического адаптера.

Алгоритм решения задачи:

1) инициируем модуль graph;

2) рисуем НЛО с помощью двух эллипсов, двух линий и двух маленьких кружочков, держим на экране, затем стираем изображение процедурой CLEARDEVICE;

3) определяем размер буфера и помещаем в него изображение;

4) организуем безусловный цикл по переменной i и рисуем звездное небо;

5) организуем цикл до тех пор, пока не будет нажата любая клавиша;

6) опять рисуем звездное небо;

7) в этом цикле помещаем изображение из буфера на экран, держим его на экране, затем стираем изображение;

8) определяем случайным образом координаты следующего изображения НЛО;

9) после нажатия любой клавиши закрываем графический режим.

```
uses crt,graph;
var drive,mode,x,y,i,r,rx,ry,lx,ly,size:integer;
    saucer:pointer;
begin
  r:=20;
  x:=r*5;
  y:=r*2;
  drive:=detect;
  initgraph(drive,mode,'c:\tp\bgi');
  setcolor(3);
  ellipse(x,y,0,360,r,(r div 3)+2);
  ellipse(x,y–4,190,357,r,r div 3);
  line(x–17,y–16,x–25,y–22);
  line(x+17,y–16,x+25,y–22);
  circle(x+25, y–25,2);
  circle(x–25, y–25,2);
  setfillstyle(1,3);
  floodfill(x+1,y+4,3);
  lx:=x–r–30;
  ly:=y–30;
  rx:=x+r+30;
  ry:=y+r div 3+30;
  size:=imagesize(lx,ly,rx,ry);
  getmem(saucer,size);
  getimage(lx,ly,rx,ry,saucer^);
  readln;
  putimage(lx,ly,saucer^,xorput);
  for i:=1 to 600 do
     putpixel(random(i),random(i),i);
```

```
  repeat
     putimage(x,y,saucer^,xorput);
     delay(150);
     putimage(x,y,saucer^,xorput);
     x:=x+random(10);
     y:=y+random(10);
  until (keypressed);
  readln;
 closegraph;
end.
```

Задача 4.7. *Построить астроиду-кривую, заданную параметрическим уравнением* $x = b \cos^3(t)$, $y = b \sin^3(t)$, *t принадлежит интервалу* $[0, 2\pi]$ *(рис. 1).*

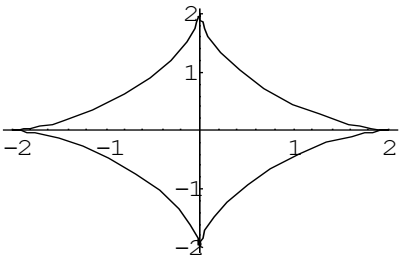


Рис. 1 — Астроида

```
uses crt,graph;
const B=200;
var drive,mode,i:integer;
   x,y,t:real;
begin
  drive:=detect;
  initgraph(drive,mode,'c:\tp\bgi');     {инициализация графического режима}
  setcolor(3);
  line(10,240,630,240);                  {ось X}
  line(320,10,320,470);                  {ось Y}
  line(630,240,610,235);                 {стрелки на оси X}
  line(630,240,610,245);
  line(320,10,315,30);                   {стрелки на оси Y}
  line(320,10,325,30);
  t:=0;
  while t<=2*pi do
  begin
       x:=b*sqr(cos(t))*cos(t);  {рассчитываем по формуле координаты точек}
       y:=b*sqr(sin(t))*sin(t);
       x:=320+x; y:=240+y;                {рисуем в центре экрана}
      {рисуем точку с координатами x, y}
       putpixel(round(x),round(y),random(15)); t:=t+0.001;
 end;
 repeat until keypressed;
```

```
  closegraph;
end.
```

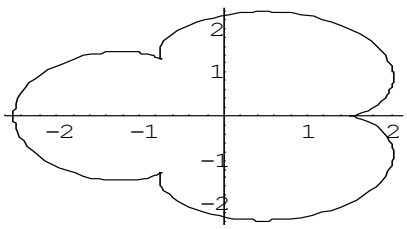
Задача 4.8. *Программа выводит окно с линиями*.
Обратите внимание, что не нужно использовать специальные процедуры для подключения графического модуля,

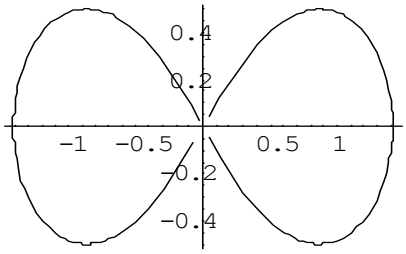
```
// Графика. Линии. Размеры окна. Заголовок окна
uses GraphABC;

begin
  Window.Title := 'Первая графическая программа';
  Line(0,0,Window.Width-1,Window.Height-1);
  Line(0,Window.Height-1,Window.Width-1,0);
end.
```

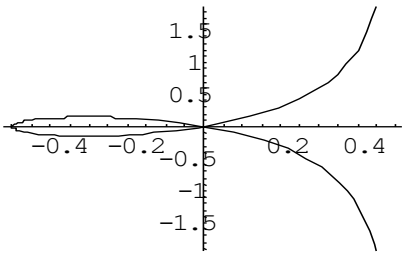
Задача 4.9. *Программа иллюстрирует обработку событий мыши.*

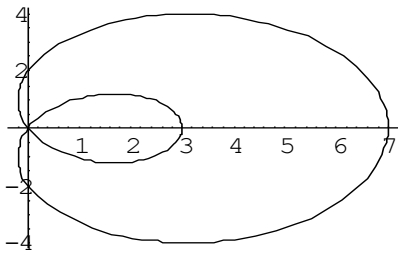
```
// Иллюстрация обработки событий мыши
uses GraphABC;

procedure MouseDown(x,y,mb: integer);
begin
  MoveTo(x,y);
end;

procedure MouseMove(x,y,mb: integer);
begin
  if mb=1 then LineTo(x,y);
end;

begin
  // Привязка обработчиков к событиям
  OnMouseDown := MouseDown;
  OnMouseMove := MouseMove
end.
```

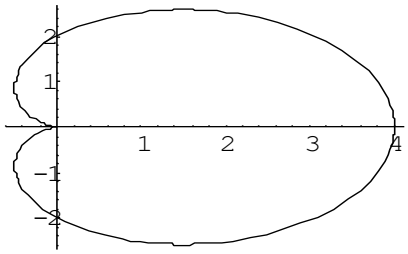
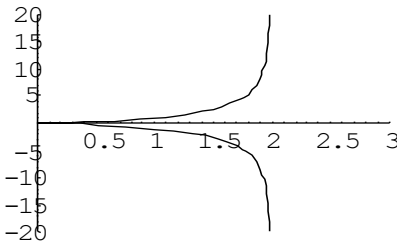
Варианты заданий лабораторной работы № 4

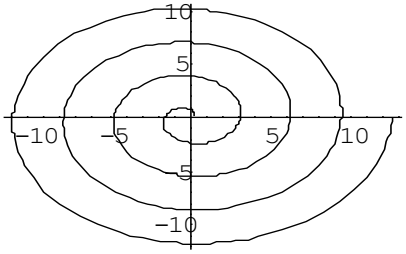
Вариант 1	Задача 1 Используйте линейные списки для хранения последовательности чисел. Опишите процедуру или функцию, которая: а) переносит в начало непустого списка его последний элемент; б) добавляет в конец списка L1 все элементы списка L2. Задача 2 Напишите программу для графической иллюстрации сортировки массива алгоритмом простого выбора. Массив изобразите в виде диаграммы — каждый элемент массива представляется в виде столбика, высота которого пропорциональна значению элемента. Визуализация сортировки сводится к показу массива после каждого перемещения элементов.
Вариант 2	Задача 1 Используйте линейные списки для хранения последовательности чисел. Опишите процедуру или функцию, которая: а) вставляет в список L за первым вхождением элемента E все элементы списка L1, если E входит в L. б) удаляет из списка L все элементы, которые есть в списке L1. Задача 2 Построить эпициклоиду — кривую, заданную параметрическим уравнениемx = (a+b) cos(t) – a cos((a+b) t/a), y = (a+b) sin(t) – a sin((a+b) t/a), a>0, b>0, b/a — целое положительное число, t принадлежит интервалу [0,2π]. 
Вариант 3	Задача 1 Используйте линейные списки для хранения последовательности чисел. Опишите процедуру, которая удаляет: а) из списка второй элемент, если такой есть; б) из непустого списка последний элемент.

<table>
<tr><td></td><td>Задача 2
Построить лемнискату — кривую, уравнение которой в полярных координатах
$\rho = a\sqrt{2\cos(2\varphi)}$.
 
</td></tr>
<tr><td>Вариант 4</td><td>Задача 1
Используйте линейные списки для хранения последовательности чисел. Опишите процедуру, которая удаляет:
а) из списка первый отрицательный элемент, если такой есть;
б) из списка все отрицательные элементы.

Задача 2
Построить строфоиду — кривую, заданную параметрическим уравнением
$x = a(t^2 - 1)/(t^2 + 1),$
$y = at\,(t^2 - 1)/(t^2 + 1),\ a>0,t$
принадлежит интервалу $[-\infty, +\infty]$.
 
</td></tr>
<tr><td>Вариант 5</td><td>Задача 1
Используйте линейные списки для хранения последовательности строк. Опишите функцию, подсчитывающую количество строк — элементов списка, которые:
а) начинаются и оканчиваются одним и тем же символом;
б) начинаются с того же символа, что и следующая строка.</td></tr>
</table>

	Задача 2 Построить кривую «улитку Паскаля» по заданному параметрическому уравнению $x = a \cos^2(t) + b \cos(t),$ $y = a \cos(t) \sin(t) + b \sin(t), a>0, b>0,$ t принадлежит интервалу $[0, 2\pi]$. Рассмотреть три случая: 1) b >= 2a (см. рис.); 2) a < b< 2a; 3) a > b. 
Вариант 6	Задача 1 Используйте линейные списки для хранения последовательности чисел. Опишите процедуру или функцию, которая для данного списка L создает список L1, содержащий те же элементы, но в обратном порядке Задача 2 Соединить конечное множество точек на плоскости замкнутой ломаной линией без самопересечений с вершинами в этих точках. (Полный перебор не делать; ответом будет порядок обхода точек плоскости.) Указание: перейти к полярным координатам и упорядочить точки по значениям угла, а для точек с одинаковым значением угла — по расстоянию до полюса.
Вариант 7	Задача 1 Используйте линейные списки для хранения последовательности вещественных чисел. Опишите процедуру или функцию, которая: а) находит среднее арифметическое элементов непустого списка; б) заменяет в списке все вхождения элемента E1 на элемент E2. Задача 2 Даны целые числа t1, t2, ... t31, — задающие график температур за март месяц. Построить график температур. Отрезки

	прямых, лежащие выше 0 градусов Цельсия и лежащие ниже 0 градусов Цельсия, должны быть окрашены в разные цвета.
Вариант 8	Задача 1 Используйте линейные списки для хранения последовательности вещественных чисел. Опишите процедуру или функцию, которая: а) меняет местами первый и последний элементы непустого списка; б) удаляет последний элемент списка. Задача 2 Построить кривую кардиоиду по заданному параметрическому уравнению $x = a \cos t (1+ \cos t)$, $y = a \sin t (1 + \cos t)$, $a>0$, t $[0, 2\prec)$. 
Вариант 9	Задача 1 Используйте линейный список для представления многочлена от переменно x, упорядоченного по степеням x. Напишите программу для дифференцирования многочлена. Задача 2 Построить кривую циссоиду по заданному параметрическому уравнению $x = a t^2/(1+t^2)$, $y = a t^3 /(1 + t^2)$, $a>0$, t $(-\mathbb{N}, \mathbb{N})$. 

Вариант 10	Задача 1 Используйте линейные списки для хранения последовательности чисел. Опишите процедуру, которая вставляет: а) новый элемент E после первого элемента непустого списка; б) новый элемент E1 за каждым вхождением элемента E. Задача 2 Построить спираль вокруг начала координат с n витками и внешним радиусом r; начальное направление спирали образует с осью x угол ↦. Параметрическое представление спирали: x = r cos t, y = r sin t, r = t/2, ↦ℂtℂ2≺ n. 
Вариант 11	Задача 1 Используйте представление последовательности строк в виде линейного списка и опишите процедуру ПЕРЕСТАНОВКА(L , i , j), меняющую местами i-ю и j-ю строки списка L. Задача 2 Напишите программу для графической иллюстрации сортировки массива алгоритмом «пузырька». Массив изобразите в виде диаграммы — каждый элемент массива представляется в виде столбика, высота которого пропорциональна значению элемента. Визуализация сортировки сводится к показу массива после каждого перемещения элементов.
Вариант 12	Задача 1 Используйте представление последовательности строк в виде линейного списка и опишите процедуру ЗАМЕНА(L , i , j), заменяющую i-ю строку списка L на копию j-й строки. Задача 2 Напишите программу для графической иллюстрации сортировки массива алгоритмом простых включений. Массив изобразите в виде диаграммы — каждый элемент массива представляется в виде столбика, высота которого пропорциональна значению элемента. Визуализация сортировки сво-

	дится к показу массива после каждого перемещения элементов.
Вариант 13	Задача 1 Используйте представление последовательности строк в виде линейного списка и опишите процедуру ДОБАВИТЬ(L , i , j), добавляющую после i-ой строки списка L копию j-й строки. Задача 2 Напишите программу для графической иллюстрации сортировки массивов алгоритмом слияния. Массивы изобразите в виде диаграммы — каждый элемент массива представляется в виде столбика, высота которого пропорциональна значению элемента. Визуализация сортировки сводится к показу массивов после каждого перемещения элементов.
Вариант 14	Задача 1 Используйте представление последовательности строк в виде линейного списка и опишите процедуру УДАЛИТЬ(L , i), удаляющую i-ю строку из списка L. Задача 2 Напишите программу, которая имитирует движение велосипеда. Указание: напишите процедуру отображения велосипеда с параметрами. В качестве параметра возьмите координаты какой-нибудь точки велосипеда (например, середина педалей — x1, y1). Назовем данную точку «центральной». Построение велосипеда ведите относительно центральной точки (например, чтобы нарисовать колесо велосипеда, можно воспользоваться следующим действием: circle(x1+50,y1,25)). Перемещение велосипеда по экрану можно получить, реализовав, например, следующий алгоритм: 1. Задаем начальные значения переменным x1 и y1. 2. Рисуем велосипед заданным цветом с данными значениями x1 и y1 (вызываем процедуру рисования велосипеда). 3. Изменяем цвет линий на цвет, соответствующий цвету фона. 4. Рисуем велосипед измененным цветом с теми же значениями x1 и y1 (вызываем процедуру рисования велосипеда). 5. Изменяем координаты x1. 6. Изменяем цвет линий на цвет, соответствующий цвету велосипеда. 7. Рисуем велосипед заданным цветом с новыми значениями x1 и y1 (вызываем процедуру рисования велосипеда).

	8. Повторяем действия 2—7.
Вариант 15	Задача 1 Используйте линейные списки для хранения последовательности чисел. Опишите процедуру или функцию, которая для данного списка L создает список L1, содержащий только положительные элементы списка. Задача 4 Написать программу, вызывающую пульсацию окружности в центре экрана. Окружность должна увеличиваться в диаметре до тех пор, пока не достигнет границ экрана, затем она начинает сжиматься. Процесс должен циклически повториться, при этом необходимо обеспечить чередование цветов при увеличении и уменьшении диаметра окружности.

ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТА

Отчеты по лабораторным работам должны быть выполнены с помощью текстового редактора Word. Требования к оформлению: шрифт основного текста Times New Roman, 12—14 пунктов, через 1,5—2 межстрочных интервала. Шрифт листинга программ — Courier New 12—14 пунктов.

Отчет к лабораторной работе должен содержать:

- Титульный лист
- Содержание
- Введение
- Основную часть
- Заключение

Пример оформления титульного листа представлен в приложении А.

Пример оформления содержания представлен в приложении Б.

Введение должно содержать краткое описание теоретического раздела, которому посвящена лабораторная работа.

Основная часть для каждого задания должна содержать:

- Точную формулировку задания.
- Описание алгоритма решения задачи.
- Описание используемых переменных и обоснование выбора типа для всех переменных.
- Блок-схему алгоритма (пример — в приложении В).
- Тестирование программы, возможно в виде скриншотов.
- Текст программы с комментариями.

СПИСОК ЛИТЕРАТУРЫ

1. Зюзьков В. М. Программирование : учеб. пособие / В. М. Зюзьков. — Томск : Эль Контент, 2013. — 186 с.

2. Немнюгин С. А. Turbo Pascal. Программирование на языке высокого уровня : учебник для вузов / С. А. Немнюгин. — 2-е изд. — СПб. : Питер, 2005. — 543 с.

3. Немнюгин С. А. Turbo Pascal : учеб. пособие для вузов / С. А. Немнюгин. — СПб. : Питер, 2003. — 491[5] с.

4. Фаронов В. В. Турбо Паскаль 7.0: Практика программирования / В. В. Фаронов. — М. Нолидж, 2003. — 415 с.

5. Немнюгин С. А. Turbo Pascal: Практикум / С. А. Немнюгин. — 2-е изд. — СПб.: Питер, 2003. — 267 с.

6. Зюзьков В М. Основы алгоритмизации и программирование на языке Паскаль : учеб.-метод. пособие / В М. Зюзьков, Е. А. Потапова, Н. Ю. Хабибулина. — Томск : ТУСУР, Кафедра КСУП, 2012. — 203 с. URL:

http://www.kcup.tusur.ru/index.php?module=mod_methodic&command=view&id=173

ПРИЛОЖЕНИЕ А
Пример оформления титульного листа

Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ОТЧЕТ

Лабораторная работа № _

по дисциплине
«Программирование»

Выполнил студент:
Иванов Иван Иванович
специальность _____

20__ г.

ПРИЛОЖЕНИЕ Б
Пример оформления содержания

СОДЕРЖАНИЕ

ПРИЛОЖЕНИЕ В
Пример оформления блок-схемы алгоритма

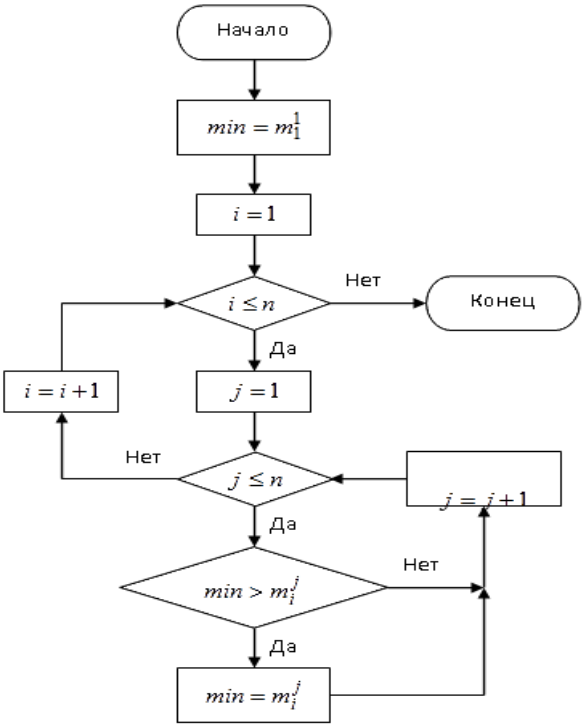


Рис. 1.1 — Блок-схема алгоритма поиска минимального элемента двухмерной матрицы

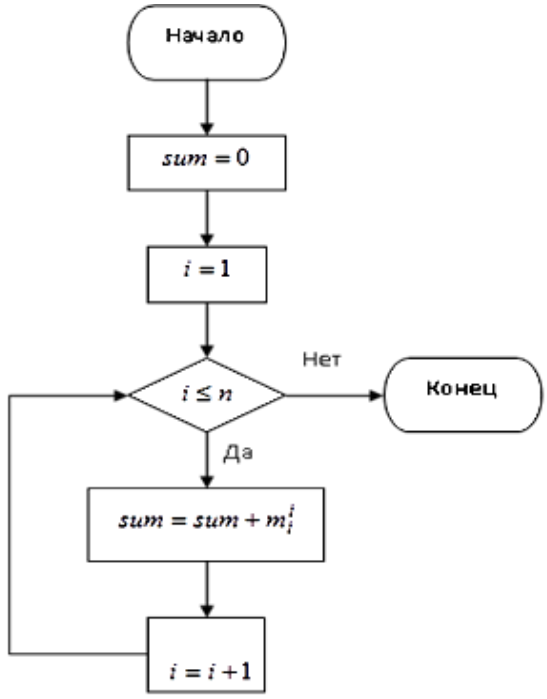


Рис. 1.2 — Блок-схема алгоритма вычисления суммы элементов главной диагонали матрицы