Песни о Паскале

Ответы на некоторые задания из секции «А слабо?»

редакция 12-B

от 2012-08-23

Аннотация

Здесь представлены часть ответов на задания «А слабо?» из книги «Песни о Паскале». Каждый рассказчик излагает событие своими словами и по-своему. Каждый инженер предлагает своё решение. Потому, сверяя свои решения с моими, не надейтесь на точное совпадение – таких совпадений не бывает!

Оглавление

Глава 5

А) Найдите ошибки в следующей программе.

```
begn

Writeln(ПрЫветик!)

end
```

Сначала проделайте это в уме, а затем на компьютере. Объясните, почему компилятор не нашел ошибки в слове «ПрЫветик».

```
Ошибки:
1. Пропущена буква в ключевом  слове begin.
2. Строковая константа не заключена в кавычки.
3. Нет точки в конце программы.
Содержимое строковых констант не проверяется.
```

Б) Будет ли работать следующая программа?

```
begin Writeln('Begin  End.')  end.
```

```
Будет.
```

В) Попытайтесь написать программу, выводящую на экран не одну, а две строки, например.

```
Без труда

Не выловишь калошу из пруда
```

Здесь нужны две процедуры печати, следующие друг за другом. Между процедурами должен стоять специальный разделитель – точка с запятой (;).

```
begin
Writeln('Без труда');
Writeln('Не выловишь калошу из пруда');
end.
```

Глава 7

А) В нашей последней программе остался маленький изъян. Со временем вы можете забыть о том, что для завершения программы надо нажать клавишу «Enter». Пусть программа сама напомнит об этом, печатая после приветствия напоминание:

```
Для завершения программы нажмите Enter
```

Внесите это изменение в программу. Или слабо?

```
begin
Writeln('------------------------');
Writeln('Мой повелитель!');
Writeln('Поздравляю тебя с написанием первой программы!');
Writeln('Твой верный слуга Паскаль');
Writeln('------------------------');
Writeln('Для завершения программы нажмите Enter');
Readln
end.
```

Б) Измените программу так, чтобы в каждой строке разместилось по два оператора. Откомпилируйте и проверьте программу в действии. Изменилось ли что-то в её поведении?

```
В поведении программы ничего не изменится.
```

Глава 8

А) Что напечатает следующая программа, если ваша любимая команда – «Спартак»?

```
Спартак – чемпион!
```

Б) Найдите и исправьте (если можно) ошибки в следующих программах.

```
const Pele = 'Эдсон Арантес ду Насименту';
begin
Writeln('Лучший футболист мира - ', Pele);
Readln
end.
```

```
var Name : string;
begin
Writeln('Как тебя зовут?');
Readln(Name);
Writeln('Здравствуй, ', Name);
Writeln('Нажми Enter'); Readln;
end.
```

```
const Pele = 'Эдсон Арантес ду Насименту';
begin
Writeln('Лучший футболист мира');
Readln(Pele); { <-- ошибка! Константу нельзя изменить! }
Writeln(Pele);
Readln
end.
```

Глава 9

А) Что напечатает следующая программа?

```
const Pele = 'Эдсон Арантес ду Насименту';
begin
Writeln('Pele = ', Pele);
end.
```

```
Pele = Эдсон Арантес ду Насименту
```

Б) А эта программа что напечатает?

```
var A, B : string;
begin
A:='123';   B:='456';
Writeln('A+B= ', A+B);
end.
```

```
A+B= 123456
```

В) Является ли следующий оператор оператором присваивания?

```
const Pele = 'Эдсон Арантес ду Насименту';
```

```
Объявление константы – это не оператор присваивания!
Объявление используется лишь в момент компиляции программы, а
присваивание, – в ходе её выполнения.
```

Д) Какие из следующих операторов «забракует» компилятор?

```
const Pele = 'Эдсон Арантес ду Насименту';
      ABBA : string = 'Музыкальный шедевр из Швеции';
var   Moscow : string;
begin
      Pele := 'Лучший футболист мира'; ← нельзя изменять константу!
      ABBA := 'Распевают частушки';    ← так можно
      Moscow := 'Столица олимпиады';
end.
```

Глава 10

А) В программах для «часового» укажите начало и конец условного оператора (то есть, первый и последний его символ, включая вложенные операторы).

```
Первый и последний символы выделены подчеркнутым жирным шрифтом
if S = 'pascal'
      then Writeln('Проходите!')
      else Writeln ('Стойте!')
```

Б) Напишите программу, которая спрашивает, идет ли дождь, и на ответ «да» выводит сообщение «А зонта-то у тебя нет!».

```
var  S : string;
begin
      Writeln('Идет ли дождь?'); Readln(S);
      if S = 'да' then Writeln('А зонта-то у тебя нет!');
      Writeln('Нажмите Enter'); Readln;
end.
```

Глава 11

А) Сколько операторов можно поместить в операторном блоке?

Количество операторов в блоке неограниченно.

Б) Найдите ошибку в этом кусочке программы, проверьте свое решение на компьютере.

```
Writeln('Что дождь? Идет?'); Readln(S);
if S = 'ага' then
      begin
            Writeln('А зонтик ты так и не купил!');
            Writeln('Сколько раз напоминать?');
      end;
else begin
            Writeln('На этот раз тебе повезло!');
      end;
```

Лишняя точка с запятой перед else

Глава 12

А) Сколько операторов можно вставить между **REPEAT** и **UNTIL**?

```
Количество вложенных операторов неограниченно.
```

Б) Будет ли проверяться условие в **UNTIL** при досрочном выходе из цикла?

```
Нет
```

В) Возьмите за основу программу «P_11_1» и сделайте из нее циклический вариант.

```
var  S : string;
begin
     repeat
          Writeln('Пароль?'); Readln(S);
          if S = 'pascal' then begin
               Writeln('Распахнуть ворота!');
               Writeln('Оркестр, музыку!');
               Writeln('Проходите!')
          end  else begin
               Writeln('Тревога!');
               Writeln('Задержать его!')
          end;
     until S='';
end.
```

Глава 13

А) Что будет напечатано в результате выполнения следующего фрагмента? Проверьте себя на компьютере.

```
S:='123';
Writeln (S='123');
```

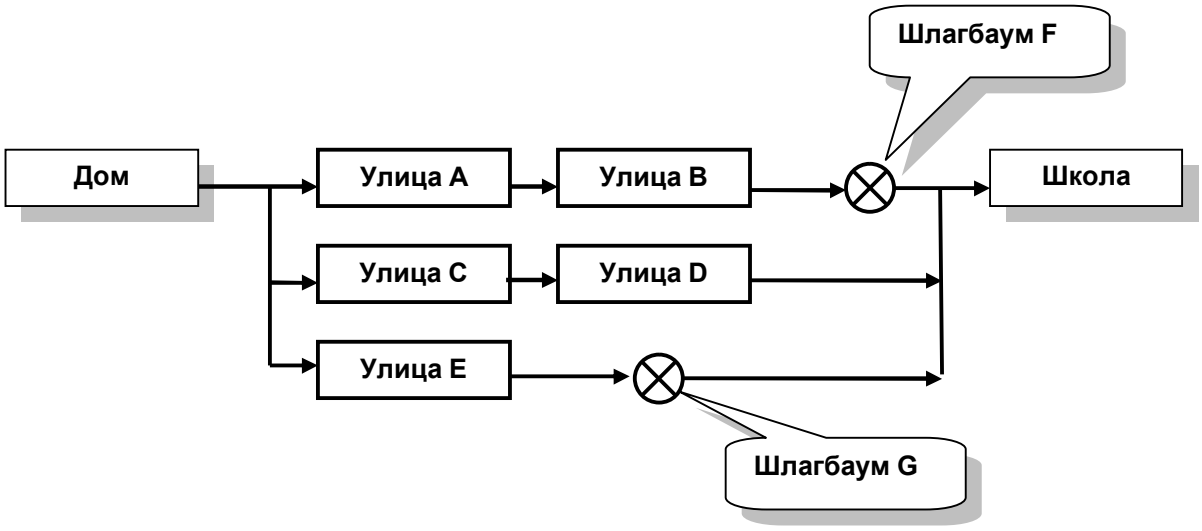
True

Б) Переведите на русский язык следующее выражение на Паскале.

```
if (S='') and (A or B) then ...
```

Если строка S пуста и A или B истины, то…

В) Напишите программу к бортовому компьютеру для плана проезда, показанного на рисунке.



Решающий оператор таков:

```
if A and B and not F or C and D or E and not G
 then Writeln('Можно ехать') ...
```

Г) Переменные **M1**, **M2** и **M3** отражают итог подбрасывания трех монет и содержат **TRUE**, если выпал «орел». Пусть программа напечатает **TRUE** для следующих случаев: 1) у всех монет выпал «орел»; 2) у всех монет выпала «решка»; 3) все три монеты упали одинаково; 4) у первой – «решка», у остальных – «орел»; 5) у первой – «орел», а две остальные упали одинаково.

```
var M1, M2, M3 : boolean;

begin
  { Ввод переменных организуйте здесь}
  Writeln(M1 and M2 and M3);
  Writeln(not M1 and not M2 and not M3);
  Writeln((M1=M2) and (M2=M3));
  Writeln(not M1 and M2 and M3);
  Writeln(M1 and (M2=M3));
end.
```

Глава 14

А) Найдите ошибки в следующих операторах и объясните их.

```
var  N, M : integer;
     S : string;
begin
     N:= '10';        { числовой переменной присваивается строка }
     S:= N + 5;       { строковой переменной присваивается число }
     M:= S - 1;       { из строковой переменной вычитается число }
     if S=N then;     { сравниваются переменные разных типов }
end.
```

Б) Перепишите программу «P_14_1», не прибегая к процедуре **Break**. В чем, по-вашему, слабость этого второго варианта?

```
var  A, B, C : integer;    { сомножители и произведение }
     R: Boolean;           { результат сравнения }
     S: string;            { сообщение для вывода на экран }
begin
     repeat
          { ввод сомножителей и произведения }
          Write('Первый сомножитель A = ');  Readln(A);
          Write('Второй сомножитель B = ');  Readln(B);
          Write('Произведение A*B = ');  Readln(C);
          if C<>0 then begin
               { проверяем правильность вычисления }
               R:= A*B=C;
               if R
                    then S:= 'Молодец, правильно!'
                    else S:= 'Повтори табл. умножения!';
               Writeln(S);
          end
     until C=0;
end.

Недостаток в том, что значение переменной C проверяется дважды.
```

В) Пусть ваша программа запросит три числа: **A**, **B** и **C**, а затем напечатает большее из трех чисел. Подсказка: примените булевы выражения вкупе с операциями сравнения. Операции сравнения в булевых выражениях надо заключать в скобки.

```
var A, B, C : integer;

begin
  Write('Введите A, B, C: '); Readln(A, B, C);
  if (A>B) and (A>C)
    then Writeln(A)
    else if (B>A) and (B>C)
      then Writeln(B)
      else Writeln(C);
  Readln;
end.
```

Г) В стене прорублено сквозное прямоугольное отверстие со сторонами **A** и **B**. Пусть ваша программа определит, пройдет ли в него кирпич с ребрами **X**, **Y**, **Z**. Соотношение между сторонами неизвестно, и программе самой следует выяснить высоту и ширину, как отверстия, так и кирпича.

```
var A, B : integer;          { стороны отверстия }
    X, Y, Z : integer;       { стороны кирпича }
    H, S : integer;          { высота и ширина отверстия, H <= S }
    Hk, Sk : integer;        { высота и ширина кирпича,  Hk <= Sk}

begin
  Write('Введит две стороны отверстия: ');  Readln(A, B);
  Write('Введите три стороны кирпича: ');  Readln(X, Y, Z);

  { Формируем высоту (H) и ширину (S) отверстия }

  if A<B
     then begin H:=A; S:=B end
     else begin H:=B; S:=A end;

  { Формируем высоту (Hk) и ширину (Sk) кирпича }

  if (X>=Y) and (X>=Z)
    then { X - длина кирпича }
           if Y<Z
                then begin Hk:=Y; Sk:=Z end
                else begin Hk:=Z; Sk:=Y end
    else if (Y>=X) and (Y>=Z)
           then { Y - длина кирпича }
                if X<Z
                     then begin Hk:=X; Sk:=Z end
                     else begin Hk:=Z; Sk:=X end
           else { Z - длина кирпича }
                if X<Y
                     then begin Hk:=X; Sk:=Y end
                     else begin Hk:=Y; Sk:=X end;

  { формируем решение }

  if (H>=Hk) and (S>=Sk)
    then Writeln ('Проходит')
    else Writeln ('Не проходит');

  Readln;

end.
```

Д) Площадь земельного участка вычисляется умножением его сторон **A** и **B**. В программу вводятся стороны двух участков (**A1**, **B1** и **A2**, **B2**), пусть она напечатает ширину и длину того участка, что больше по площади. Ширина должна быть не больше длины.

```
var A1, B1, A2, B2 : integer;
    A, B : integer; { ширина и длина }

begin
  Write('A1= '); Readln(A1);
  Write('B1= '); Readln(B1);
  Write('A2= '); Readln(A2);
  Write('B2= '); Readln(B2);
  if A1*B1 > A2*B2
    then  if A1<B1
            then begin
                   A:=A1; B:=B1
                 end
            else begin
                   A:=B1; B:=A1
                 end
    else  if A2<B2
            then begin
                   A:=A2; B:=B2
                 end
            else begin
                   A:=B2; B:=A2
                 end;
  Writeln('Ширина A= ',A,' Длина B= ',B);
  Readln;
end.
```

Глава 15

А) В каких пределах будут генерироваться числа следующими выражениями:

```
10+Random(10);                  { 10...19 }
Random(20);                     ( 0...19 }
Random(10) + Random(10);        { 0...18 }
```

Г) Найдите способ сформировать ряд случайных булевых значений (**False**, **True**), напечатайте 20 из них. Подсказка: булевы значения можно получить, сравнивая два случайных числа.

```
var i : integer;
begin
  i:=0;
  repeat
    Writeln(Random(2) = Random(2));
    i:=i+1;
  until i=20;
  Readln;
end.
```

Д) Сгенерируйте два случайных числа (в диапазоне от 1 до 10) так, чтобы они не совпадали. Сделайте то же самое для трех чисел.

```
var N1, N2, N3 : integer;

begin
  Randomize;
  N1:= 1+ Random(10);
  repeat
    N2:= 1+ Random(10);
  until N1<>N2;
  repeat
    N3:= 1+ Random(10);
  until (N3<>N1) and (N3<>N2);
  Writeln(N1,' ', N2,' ', N3);
  Readln;
end.
```

Глава 16

А) Какой ответ будет выпадать чаще других, если условием в операторе **CASE** нашей программы поставить выражение **Random(100)**?

«Не знаю, я не местный» (ветвь ELSE).

Б) Напишите программу, которая бы запрашивала номер дня недели, и в ответ печатала бы название этого дня («понедельник», «вторник» и так далее).

```
Оператор выбора будет таким:
     case Day of
          1: S:='Понедельник';
          2: S:='Вторник';
          . . .
          7: S:='Воскресенье';
          else S:='Ошибка!';
     end;
```

В) Пусть пользователь введет число – свой возраст в годах. Ваша программа должна напечатать фразу: «Вам столько-то лет» с правильным окончанием, например: «Вам 20 *лет*», или «Вам 34 *года*», или «Вам 41 *год*». Подсказка: надо определить последнюю цифру года операцией **MOD 10**. Некоторые числа выпадают из общего правила, их надо проверить особо (например, 11, 12, 13, 14).

Первый вариант программы:

```
var Y : integer;
    S : string;

begin
  repeat
    Write('Сколько лет? '); Readln(Y);
    case Y of
      11: S:='лет';
      12: S:='лет';
      13: S:='лет';
      14: S:='лет';
      else case Y mod 10 of
             1: S:='год';
             2: S:='года';
             3: S:='года';
             4: S:='года';
             else S:='лет';
           end;
    end;
    Writeln ('Вам ', Y,' '+S);
  until Y=0;
end.
```

Второй вариант программы:

```
var Y : integer;
    S : string;

begin
  repeat
    Write(' Сколько лет?  '); Readln(Y);
    case Y of
      11..14: S:='лет';
      else case Y mod 10 of
             1: S:='год';
             2..4: S:='года';
             else S:='лет';
           end;
    end;
    Writeln ('Вам ', Y,' '+S);
  until Y=0;
end.
```

Г) Пользователь вводит число – номер месяца от 1 до 12, а программа должна сообщить соответствующее ему время года: зима, весна, лето, осень.

```
var N : integer;

begin
  Write('N= '); Readln(N);
  case N of
    1,2,12 : Writeln(' Зима ');
    3,4, 5 : Writeln(' Весна ');
    6,7, 8 : Writeln(' Лето ');
    9,10,11: Writeln(' Осень ');
    else Writeln(' Ошибка! ');
  end;
  Readln;
end.
```

Д) Танк в компьютерной игрушке может двигаться в одном из четырех направлений, обозначим их числами: 1 – север, 2 – восток, 3 – юг, 4 – запад. Направление движения изменяется тремя командами: 1 – поворот направо, 2 – поворот налево, 3 – поворот кругом. Пользователь вводит начальное направление движения, а затем ряд команд. Программа должна определять и печатать всякий раз новое направление. Выход из цикла – команда 0.

```
var
     Direct, Command : integer;

begin
  Write('Направление(1-4)= '); Readln(Direct);
  repeat
    Write('Команда (1-3)= '); Readln(Command);
    case Command of
      1 : case Direct of          { направо }
            1: Direct:= 2;
            2: Direct:= 3;
            3: Direct:= 4;
            4: Direct:= 1;
          end;
      2 : case Direct of          { налево }
            1: Direct:= 4;
            2: Direct:= 1;
            3: Direct:= 2;
            4: Direct:= 3;
          end;
      3 : case Direct of          { кругом }
            1: Direct:= 3;
            2: Direct:= 4;
            3: Direct:= 1;
            4: Direct:= 2;
          end;
    end;
    Writeln('Направление= ', Direct);
  until Command=0;
end.
```

Е) Исходное состояние шахматных фигур известно всякому (если вы исключение из правила, ознакомьтесь с основами шахмат). Пользователь в цикле вводит число, по которому программа печатает название фигуры, стоящей на соответствующей вертикали шахматной доски (от 1 до 8). Ноль служит для выхода из цикла, а на все прочие числа программа сообщает об ошибке.

```
var N: integer;
begin
  repeat
    Write('N= '); Readln(N);
    case N of
      1,8: Writeln('Ладья');
      2,7: Writeln('Конь');
      3,6: Writeln('Слон');
        4: Writeln('Ферзь');
        5: Writeln('Король');
        0: Writeln('До встречи!');
    else
        Writeln('Ошибка!');
    end
  until N=0
end.
```

Ж) Программа запрашивает в цикле два числа: вертикаль и горизонталь шахматной доски (числа от 1 до 8), а затем печатает цвет клетки на их пересечении. Если хотя бы одно из чисел равно нулю, цикл завершается. Если числа выходят за указанные пределы, сообщает об ошибке и повторяет запрос чисел.

Примечание: на пересечении 1-й строки и 1-го столбца находится белая клетка.

```
var X, Y: integer;
begin
  repeat
    Y:=1;
    Write('X= '); Readln(X);
    if X=0 then break;
    if (X>0) and (X<9) then begin
      Write('Y= '); Readln(Y);
      if Y=0 then break;
      if (Y>0) and (Y<9) then begin
        if ((X+Y) mod 2)=0
          then Writeln('Белая клетка')
          else Writeln('Черая клетка')
      end else begin
        Writeln('Ошибка в Y!');
      end
    end else begin
      Writeln('Ошибка в X!');
    end
  until (X=0) or (Y=0);
  Writeln('До встречи!');
end.
```

Глава 17

А) Дайте ученику возможность отказаться от сдачи экзамена. Признаком отказа будет ввод числа ноль в качестве ответа. В этом случае надо досрочно выйти из цикла и обойти выставляющий оценку оператор.

```
var  A, B, C : integer;    { сомножители и произведение }
     Q, E : integer; { счетчик вопросов и счетчик ошибок }
     S: string;
begin
     Randomize;
     E:= 0; { обнуляем счетчики ошибок }
     for Q:= 1 to 15 do begin  { 15 вопросов }
          A:= 1+ Random(10);    B:= 1+ Random(10);
          Write(Q,') Сколько будет ', A,' x ',B, ' ? ');
          Readln(C);
          if C=0 then break;
          { Если ответ неверный, увеличиваем счетчик ошибок }
          if A*B <> C then E:= E+1;
     end;  { цикл (и блок) завершается здесь}
     if C<>0 then begin
          case E of  { выставляем оценку }
               0:    S:='Отлично!';
               1,2:  S:='Хорошо';
               3..5:S:='Удовлетворительно';
               else  S:='Ну оччччень плохо!';
          end;
          Writeln(S, '  Нажмите Enter'); Readln;
     end
end.
```

Б) Напишите программу, которая по введенному числу дает заключение о том, какому дню недели оно соответствует – рабочему (1-5) или выходному (6,7).

```
Оператор выбора будет таким:
     case Day of
          1..5: S:='Рабочий';
          6,7: S:='Выходной';
          else S:='Ошибка!';
     end;
```

В) Напишите программу, которая, запросив число **N**, печатала бы числа от 1 до **N** в обратном порядке, например:

```
N = 3
3
2
1
```

```
for i:=1 to N do Writeln(N+1-i);
```

Г) Существует вариант оператора **FOR...**, где счетчик цикла не наращивается, а уменьшается, этот оператор записывается так:

FOR N:= начальное_значение **DOWNTO** конечное_значение **DO** оператор

Ключевое слово **DOWNTO** задает счет в обратном порядке (**DOWN** – «вниз»); при этом предполагается, что начальное значение счетчика больше или равно конечному, иначе цикл не выполнится ни разу. Воспользуйтесь этим оператором для решения предыдущей задачи (**В**).

```
for i:=N downto 1 do Writeln(i);
```

Д) Пусть программа запросит два числа **N** и **M**, а затем вычислит их произведение без использования операции умножения (*). Подсказка: организуйте цикл суммирования **N** раз числа **M**.

```
var  N, M : integer;
     i, sum : integer;
begin
     Write('Введите через пробел N и M '); Readln(N, M);
     sum:=0;
     for i:=1 to N do sum:= sum + M;
     Write(N,' * ',M,' = ',sum); Readln;
end.
```

Е) Напишите программу, вычисляющую сумму чисел от 1 до **N**, где **N** – число, вводимое пользователем.

```
var  N : integer;
     i, sum : integer;
begin
     Write('Введите N '); Readln(N);
     sum:=0;
     for i:=1 to N do sum:= sum + i;
     Write('C = ',sum); Readln;
end.
```

Ж) Напишите программу, вычисляющую сумму только тех чисел от 1 до **N**, которые делятся либо на три, либо на пять (см. решение для задачи Е).

```
     for i:=1 to N do if (i mod 3 = 0) or (i mod 5 = 0)
          then sum:= sum + i;
```

И) Платный участок трассы протянулся с **P1** до **P2** километра (**P1**<**P2**). А пост ГАИ размещен на километре **M**. Расположен ли этот пост на платном участке трассы? Пусть ваша программа разберется с этим.

```
var  P1, P2, M : integer;
begin
     Write('Введите через пробел P1, P2, M '); Readln(P1,P2,M);
     if (M>=P1) and (M<=P2)
          then Writeln('На платной')
          else Writeln('На бесплатной')
end.
```

К) Дорожная служба запланировала ремонт трассы на участке с **R1** по **R2** (**R1**<**R2**). В сочетании с условием предыдущей задачи ваша программа должна определить:

- Будут ли ремонтировать весь платный участок **P1-P2** ?
- Будут ли ремонтировать хотя бы часть платного участка **P1-P2** ? Если да, то определить длину ремонтируемой платной части.
- Будут ли ремонтировать хотя бы часть бесплатного участка? Если да, то определить длину ремонтируемой бесплатной части.

```
{ Это решение предложено моим читателем М.В.Артищевым }

var
  P1,P2   : integer; { границы платного }
  R1,R2   : integer; { границы ремонтируемого }
  RPL,RCLL: integer; { длины ремонтируемых платного и бесплатного }

 begin
  Write('Платный участок: '); Readln(P1, P2);
  Write('Ремонтируемый участок: '); Readln(R1, R2);
  Write('Будут ли ремонтировать ВЕСЬ платный участок? ');
  if (P1>=R1) and (P2<=R2)
    then Writeln(' - Да')
    else Writeln(' - Нет');
 {
 Будут ли ремонтировать хотя бы часть платного участка P1-P2 ?
 Если да, то определить длину ремонтируемой платной части.
 }
  Write('Будут ли ремонтировать ЧАСТЬ платного участка? ');
  if (R2<=P1) or (R1>=P2)
    then
      begin
        Writeln(' - Нет');
        RPL:=0;
      end
    else
      begin
        Writeln(' - Да');
        if P2<R2
          then RPL:=P2
          else RPL:=R2;
        if P1>R1
          then RPL:=RPL-P1
          else RPL:=RPL-R1;
        Writeln('Длина ремонтируемого платного участка ',RPL);
      end;
```

```
{
Будут ли ремонтировать хотя бы часть бесплатного участка?
Если да, то определить длину ремонтируемой бесплатной части.
}
 Write('Будут ли ремонтировать хотя бы часть бесплатного участка? ');
 if (R1>=P1) and (R2<=P2)
   then Writeln(' - Нет')
   else
     begin
       Writeln(' - Да');
       RCLL:=R2-R1-RPL;
       Writeln('Длина ремонтируемого бесплатного участка ',RCLL);
     end;
 Readln;
end.
```

Глава 18

А) Напишите программу для подсчета букв «А» во введенной пользователем строке.

```
N:=0;
for i:=1 to Length(S) do if S[i]='A' then N:=N+1;
```

Б) Напишите программу, меняющую символы «А» строки на символы «Б». Подсказка: присвоение значения некоторому символу строки делается оператором вида **S[i]:=**…

```
for i:=1 to Length(S) do if S[i]='A' then S[i]:='B';
```

Г) Записи телефонных номеров обычно содержат дополнительные символы (скобки, черточки, пробелы), - чтобы удобней запоминались. Например: 8 (123) 45-67-89. Предположим, что пользователь их так и вводит. Пусть ваша программа преобразит строку с номером, удалив из нее все символы, кроме цифр. Например, после ввода указанного выше номера она должна напечатать: **8123456789**.

```
var   i: integer;
      S, R: string;
      c: char;

begin
  S:= '(8)123-45-67';
  R:= '';
  for i:=1 to Length(S) do begin
    c:= S[i];
    if (c='0')or(c='1')or(c='2')or(c='3')or(c='4')or(c='5')
       or(c='6')or(c='7')or(c='8')or(c='9')
       then R:= R+c;
  end;
  Writeln(R);
  Readln;
end.
```

Д) Пусть ваша программа напечатает введенную пользователем строку вразрядку, добавляя пробел после каждого символа, например: **'Pascal' → 'P a s c a l'**.

```
var   i: integer;
      S, R: string;
begin
  Readln(S);
  R:='';
  for i:=1 to Length(S) do R:=R+S[i]+' ';
  Writeln(R);
  Readln;
end.
```

Глава 19

А) Напишите еще одну версию процедуры **Pause**, выводящую сообщение либо на русском, либо на английском языке. Параметр этой процедуры должен быть булевым и работать она должна так.

```
Pause(true);      { печатается «Нажмите Enter…» }
Pause(false);     { печатается «Press Enter…» }
```

```
procedure Pause(lang: boolean);
begin
  if lang
   then Writeln('Нажмите Enter')
   else Writeln('Press Enter');
  Readln;
end;

begin
  Pause(true);
  Pause(false)
end.
```

Б) Напишите и испытайте процедуру (назовем её **Line** – «линия»), печатающую строку заданной длины, составленную из звездочек, например.

```
Line(3);    { печатает «***» }
Line(7);    { печатает «*******» }
```

```
procedure Line(len: integer);
var i: integer;
begin
  for i:=1 to len do Write('*');
  Writeln
end;

begin
  Line(3);
  Line(7);
  Readln;
end.
```

В) Напишите процедуру для очистки экрана, она может пригодиться вам в будущем. Подсказка: можно напечатать несколько десятков пустых строк (не менее 25, - зависит от настройки консольного окна).

```
const CScreen = 25; { количество строк в консольном окне }

procedure ClearScreen;
var i: integer;
begin
  for i:= 1 to CScreen do Writeln;
end;

begin
  Writeln('************'); Readln;
  ClearScreen; Readln;
end.
```

Г) Напишите процедуру, принимающую два числа (два параметра), и печатающую два числа: их сумму, и их разность.

```
procedure Arithm (a, b : integer);
begin
  Writeln('A+B= ',a+b,' A-B= ',a-b);
end;

begin
  Arithm(25, 15);
  Arithm(10, 20);
  Readln;
end.
```

Д) Пользователь вводит строку с телефонным номером (только цифры), количество цифр заранее неизвестно. Ваша программа должна дополнить номер дефисами, разбивающими его на триады, т.е. по три цифры двумя способами:

- начиная с первых цифр, например 112-345-1;
- начиная с последних цифр, например 1-123-451.

```
var S, T : string;
    i, n: integer;
begin
  Write('Номер: '); Readln(S);
  T:='';
  for i:= 1 to Length(S) do begin
    T:=T+S[i];
    if (i mod 3 = 0) and (i<Length(S)) then T:=T+'-';
  end;
  Writeln(T);
  T:=''; n:=1;
  for i:= Length(S) downto 1 do begin
    T:=S[i]+T;
    if (n mod 3 = 0) and (i>1) then T:='-'+T;
    n:= n+1;
  end;
  Writeln(T);
```

```
  Readln;
end.
```

Е) Почтальон разносит газеты по улице, состоящей из **N** домов. Четные и нечетные номера расположены по разные стороны улицы. В здравом уме почтальон не рискует лишний раз переходить её. Ваша программа должна напечатать последовательность номеров, по которым будут разнесена почта, когда почтальон начинает работу:

- с первого дома;
- со второго дома;
- с N-го (последнего) дома.

```
var
  i, k, N: integer;
begin
  Write('Количество домов: '); Readln(N);

  Writeln('С 1-го:');
  for i:=1 to N do
     if i mod 2 <>0 then Write(i, ' ');
  for i:=N downto 2 do
     if i mod 2 =0 then Write(i, ' ');
  Writeln;

  Writeln('Со 2-го:');
  for i:=2 to N do
     if i mod 2 =0 then Write(i, ' ');
  for i:=N downto 1 do
     if i mod 2 <>0 then Write(i, ' ');
  Writeln;

  Writeln('С последнего:');
  k:= N mod 2; { k=0, если N четное }
  for i:=N downto 1 do
     if i mod 2 = k then Write(i, ' ');
  for i:=1 to N do
     if i mod 2 <> k then Write(i, ' ');
  Writeln;
  Readln;
end.
```

Глава 20

А) В 17-й главе нами создан экзаменатор, проверяющий знания таблицы умножения. Переработайте программу P_17_1 так, чтобы операторы, выставляющие оценку, были выделены в процедуру, принимающую один параметр – количество допущенных ошибок.

```
var  A, B, C : integer;     { сомножители и произведение }
     Q, E : integer;        { счетчик вопросов и счетчик ошибок }

procedure Arbitr(errors : integer);
var  S: string;
begin
     case errors of  { выставляем оценку }
          0:    S:='Отлично!';
          1,2:  S:='Хорошо';
          3..5: S:='Удовлетворительно';
          else  S:='Ну оччччень плохо!';
     end;
     Writeln(S, '  Нажмите Enter');
end;

begin
     Randomize;
     E:= 0; { обнуляем счетчики ошибок }
     for Q:= 1 to 15 do begin  { 15 вопросов }
          A:= 1+ Random(10);    B:= 1+ Random(10);
          Write(Q,') Сколько будет ', A,' x ',B, ' ? ');
          Readln(C);
          if A*B <> C then E:= E+1;
     end;
     Arbitr(E);
     Readln;
end.
```

Б) Создайте процедуру, печатающую все числа, кроме единицы, на которые без остатка делится число **N**, где **N** – параметр процедуры. Напишите программу для проверки этой процедуры.

```
procedure Test(N : integer);
var  i: integer;
begin
     for i:=2 to N div 2 do
          if N mod I = 0 then Writeln(i);
     Readln;
end;

begin
     Test(12);
     Test(18);
end.
```

В) Два сотрудника подали своему начальнику заявления на отпуск. Первый попросил отпустить его с **A1** по **B1** день (дни отсчитываются с начала года), второй – с **A2** по **B2** день. Считаем, что **A1<B1** и **A2<B2**. Однако дело требует, чтобы один из сотрудников находился на работе. Мало того, при смене отдыхающих необходимо не менее 3-х дней их совместной работы – для передачи дел.

Напишите процедуру, принимающую четыре указанных выше параметра, и печатающую заключение о том, удовлетворяют ли заявления требованиям начальника.

```
procedure Test(A1, B1, A2, B2 : integer);
var S: string;
begin
  if B2>B1 then begin
    if A2-B1 > 3
      then S:= 'Отпускаем'
      else S:= 'Не отпускаем';
  end else begin
    if A1-B2 > 3
      then S:= 'Отпускаем'
      else S:= 'Не отпускаем';
  end;
  Writeln(S);
end;

var  A1, B1, A2, B2: integer;
begin
  Write('Введите : A1, B1, A2, B2 : ');
  Readln(A1, B1, A2, B2);
  Test(A1, B1, A2, B2);
  Readln;
end.
```

Г) Подойдя к перекрестку, пешеход думает о том, переходить ли ему улицу, или остановиться. На решение влияет характер пешехода и еще два фактора: сигнал светофора и близость опасно движущегося транспорта. Напишите программу с процедурой, которая принимает и печатает решение в зависимости от переданных в неё трех параметров, а именно.

- Параметр **A = true**, если горит зеленый;
- Параметр **B = true**, если поблизости опасно движется транспорт;
- Параметр **C** – это число, определяющее характер пешехода так:

1 - послушный и осторожный – учитывает светофор и опасность;
2 - послушный, но беспечный – смотрит только на светофор;
3 - хитрый вольнодумец – идет только на красный, если это ничем не грозит;
4 - непримиримый вольнодумец – идет на красный, невзирая на опасность;
5 - «экстремал» – идет только на красный, но так, чтобы грозила опасность;
6 - «безбашенный» – идет, несмотря ни на что;
7 - запуганный – никогда не идет через дорогу, а ищет подземный переход.

```
procedure Solve(A, B : boolean; C: integer);
var r: boolean;
begin
  case C of
    1: { - послушный и осторожный }
        r:= A and not B;
    2: { - послушный, но беспечный }
        r:= A;
    3: { - хитрый вольнодумец }
        r:= not A and not B;
    4: { - непримиримый вольнодумец }
        r:= not A;
    5: { - экстремал }
        r:= not A and B;
    6: { - безбашенный }
        r:= true;
    7: { - запуганый }
        r:= false;
  else
    r:= false;
  end;
  if r
    then Writeln('Вперед!')
    else Writeln('Стой!');
end;

var X: char;
    A, B : boolean;
    C : integer;

begin
  repeat
    Write('Характер пешехода (1..7)= '); Readln(C);
    if C=0 then break;
    Write('Светофор зеленый? '); Readln(X);
    A:= X='y';
    Write('Есть опасность от транспорта? '); Readln(X);
    B:= X='y';
    Solve(A, B, C);
  until false
end.
```

Глава 23

А) Напишите функцию для проверки наличия буквы в заданной строке. Функция возвращает значение **TRUE**, если в строке есть хоть одна эта буква, и **FALSE** в противном случае. Напишите программу для проверки функции. Или слабо?

```
function TestChar(str: string; sym: char): boolean;
var i: integer;
begin
  TestChar:= false;
  for i:=1 to Length(str) do if str[i]=sym then begin
    TestChar:= true;
    Break
  end;
end;

begin
  Writeln(TestChar('Pascal', 's'));
  Writeln(TestChar('Pascal', 'L'));
  Readln;
end.
```

Б) Напишите функцию (и программу для её проверки), принимающую число и возвращающую строку: слово «четное» или «нечетное» в зависимости от четности или нечетности параметра. Подсказка: для проверки четности числа **N** надо проверить остаток от его деления на два:

```
      if (N mod 2) = 0 then { четное }
```

```
function TestNumber(Number: integer): boolean;
begin
  TestNumber:= (Number mod 2) = 0;
end;

begin
  Writeln(TestNumber(10));
  Writeln(TestNumber(11));
  Readln;
end.
```

Глава 24

Б) Предположим, вы пятикратно зашифровали строку. Можно ли расшифровать её? И как это сделать?

```
Расшифровать пять раз
```

В) Для введенной пользователем строки напечатать позиции всех встречающихся в ней символов, кроме пробелов, в алфавитном порядке. Для символов, которые встречаются несколько раз, напечатать позиции в одной строке.

```
var  S: string;
     c: char;
     i: integer;
     flag: boolean; { признак для печати очередной строки }
begin
  Write('S = ');  Readln(S);
  { Char(33) - первый символ после пробела }
  for c:=Char(33) to Char(255) do begin
    flag:= false;
    for i:=1 to Length(S) do if c=S[i] then begin
      if not flag then Write(c, ' - ');
      Write(i,' ');
      flag:= true;
    end;
    if flag then Writeln;
  end;
  Readln;
end.
```

Г) Для введенной пользователем строки напечатать позиции всех встречающихся в ней символов, кроме пробелов, в порядке их следования в строке.

```
var  S: string;
     c: char;
     i, j: integer;
begin
  Write('S = ');  Readln(S);
  for i:=1 to Length(S)  do begin
    c:= S[i];
    if c<>Char(32) then begin  { Char(32) - пробел }
      Write(c, ' - ');
      for j:=i to Length(S) do if c=S[j] then begin
        Write(j,' ');
        S[j]:= Char(32);  { затираем символ пробелом }
      end;
      Writeln;
    end;
  end;
  Readln;
end.
```

Д) Строки текстовых файлов порой содержат управляющие символы, например символ горизонтальной табуляции (код 9). Шифрование этих символов нарушит структуру файла. Исправьте функции **Crypt** и **DeCrypt** так, чтобы они не изменяли символы, коды которых меньше 32.

```
{ Шифрование одного символа }

function CryptChar(arg: char): char;
var x: integer;
begin
  CryptChar:=arg;
  if Ord(arg)>=32 then begin  { кроме управляющих символов! }
    x:= Ord(arg)+ CKey;
    if x>255 then x:= x-256+32;
    CryptChar:= Char(x);
  end;
end;

  { Дешифрование одного символа }

function DeCryptChar(arg: char): char;
var x: integer;
begin
  DeCryptChar:=arg;
  if Ord(arg)>=32 then begin  { кроме управляющих символов! }
    x:= Ord(arg)- CKey;
    if x<32 then x:= x+256-32;
    DeCryptChar:= Char(x);
  end;
end;
```

Глава 25

А) Можно ли связать текстовую переменную **F** с файлом оператором присваивания?

```
F := 'c:\autoexec.bat';
```

Нельзя. Проверьте на практике.

Б) Напишите программу для вывода на экран файла, имя которого вы будете вводить с клавиатуры.

```
var F: text;
    S: string;

begin
  Write('Имя файла: '); Readln(S);
  Assign(F, S);
  Reset(F);
  while not Eof(F) do begin
    Readln(F, S);
    Writeln(S);
  end;
  Readln;
end.
```

В) Напишите три функции для подсчета:

- количества строк в файле;
- количества видимых символов в файле;
- количества всех символов файла (объем файла).

Функции принимают один параметр – ссылку на файловую переменную. Напишите программу, подсчитывающую упомянутые выше характеристики файла.

```
var F_in: text;

function CalcLines(var F: Text): integer;
var n: integer;
begin
  n:=0;
  Reset(F);
  while not Eof(F) do begin
    Readln(F);
    n:= n+1;          { количество строк }
  end;
  CalcLines:=n;
end;
```

```
function CalcChars(var F: Text): integer;
var n: integer;
    S: string;
begin
  n:=0;
  Reset(F);
  while not Eof(F) do begin
    Readln(F, S);
    n:= n+Length(S);  { количество видимых символов }
  end;
  CalcChars:=n;
end;


function CalcSize(var F: Text): integer;
var n: integer;
    S: string;
begin
  n:=0;
  Reset(F);
  while not Eof(F) do begin
    Readln(F, S);
    n:= n+Length(S)+2;  { символов вместе с CR+LF}
  end;
  CalcSize:=n;
end;


begin
  Assign(F_in, 'Test.in');
  Writeln('Строк: ', CalcLines(F_in));
  Writeln('Видимых символов: ', CalcChars(F_in));
  Writeln('Объем файла: ', CalcSize(F_in));
  Close(F);
  Readln;
end.
```

Г) Объявите две файловые переменные, свяжите их с одним и тем же файлом, а затем откройте через обе переменные. Вызовет ли это ошибку? Объясните результат, исходя из здравого смысла.

```
{ Ошибки не происходит,
 поскольку для чтения файл может открываться без ограничений }

var F1, F2: text;
begin
  Assign(F1, 'c:\autoexec.bat');
  Assign(F2, 'c:\autoexec.bat');
  Reset(F1);  Reset(F2);
  Writeln('OK!');
  Readln;
end.
```

E) Напишите процедуру для вывода на экран **n**-й строки файла, где **n** – параметр процедуры. Воспользовавшись этой процедурой, напишите программу для распечатки строк файла в обратном порядке. Подсказка: предварительно посчитайте количество строк в файле.

```
var F_in: text;
    K:integer;

function CalcLines(var F: Text): integer;
var n: integer;
begin
  n:=0;
  Reset(F);
  while not Eof(F) do begin
    Readln(F);
    n:= n+1;          { количество строк }
  end;
  CalcLines:=n;
end;

procedure Expo(var F: Text; n: integer);
var S: string;
begin
  Reset(F);
  while (n>1) and not Eof(F) do begin
    Readln(F);
    n:= n-1;
  end;
  Readln(F, S);
  Writeln(S);
end;

begin
  Assign(F_in, 'Test.in');
  K:= CalcLines(F_in);    { количество строк }
  while K>0 do begin
    Expo(F_in, K);  K:= K-1;
  end;
  Readln;
end.
```

Глава 26

А) Напишите программу, которая создает файл, печатает в него несколько строк с числами, а затем выводит этот файл на экран. Воспользуйтесь одной файловой переменной.

```
var F: text;
    i: integer;
    s: string;

begin
  Assign(F, 'Test.txt');
  Rewrite(F);
  for i:=1 to 10 do Writeln(F, i);
  Close(F);
  Reset(F);
  while not Eof(F) do begin
    Readln(F, S);
    Writeln(S);
  end;
  Readln;
end.
```

Б) Напишите программу для нумерации строк файла. Строки исходного файла должны копироваться в конечный файл с добавлением перед каждой строкой её номера, например.

Исходный файл:

```
В лесу родилась елочка,
В лесу она росла.
Зимой и летом стройная,
Зеленая была.
```

Конечный файл:

```
1
В лесу родилась елочка,
2
В лесу она росла.
3
Зимой и летом стройная,
4
Зеленая была.
```

```
var F1, F2: text;
    N: integer;
    s: string;

begin
  Assign(F1, 'Test.in');  Reset(F1);
  Assign(F2, 'Test.out'); Rewrite(F2);
  N:=1;
  while not Eof(F1) do begin
    Readln(F1, S);
    Writeln(F2,N);
    Writeln(F2,S);
    N:= N+1;
  end;
  Close(F1);  Close(F2);
end.
```

В) Скопировать один файл в другой с перестановкой местами четных и нечетных строк.

```
var F1, F2: text;
    S1, S2: string;

begin
  Assign(F1, 'Test.in');  Reset(F1);
  Assign(F2, 'Test.out'); Rewrite(F2);
  while not Eof(F1) do begin
    Readln(F1, S1);
    if not Eof(F1) then begin
      Readln(F1, S2);
      Writeln(F2,S2);
    end;
    Writeln(F2,S1);
  end;
  Close(F1);  Close(F2);
  Writeln('OK');
  Readln;
end.
```

Г) Для передачи по интернету секретного текстового файла создать из него два других: в первый записать нечетные строки исходного файла, а во второй – четные.

```
var F0, F1, F2 : Text;
    n: integer;
    S: string;

begin
  Assign(F0,'Test.in');
  Assign(F1,'Test1.out');
  Assign(F2,'Test2.out');
  Reset(F0);
  Rewrite(F1); Rewrite(F2);
  n:=0;
```

```
while not Eof(F0) do begin
    Readln(F0, S);
    case n of
      0: Writeln(F1,S);
      1: Writeln(F2,S);
    end;
    n:= (n+1) mod 2;
  end;
  Close(F0);
  Close(F1); Close(F2);
  Writeln('OK');  Readln;
end.
```

Д) Создать программу для объединения двух файлов: из первого составляются нечетные строки конечного файла, а из второго – четные (см. условие предыдущей задачи).

```
var F0, F1, F2 : Text;
    flag: boolean;
    S: string;

begin
  Assign(F0,'Test.out');
  Assign(F1,'Test1.in');
  Assign(F2,'Test2.in');
  Rewrite(F0);
  Reset(F1); Reset(F2);
  flag:=false;
  while not Eof(F1) or not Eof(F2) do begin
    case flag of
      false: if not Eof(F1) then Readln(F1,S);
      true:  if not Eof(F2) then Readln(F2,S);
    end;
    Writeln(F0, S);
    flag:= not flag;
  end;
  Close(F0);
  Close(F1); Close(F2);
  Writeln('OK'); Readln;
end.
```

Глава 27

В) Некоторые строки исходного файла содержат круглые скобки (это может быть программа или математические выкладки – неважно). Ваша программа должна распечатать строки, где скобки расставлены неверно, вот примеры.

```
2+3          – правильно;
(2+3         – ошибка;
()2+3()      – правильно;
))2+3((      – ошибка.
```

Рекомендация: для исследования строки напишите булеву функцию **Check**, возвращающую **TRUE**, если скобки расставлены без ошибок.

```
function Check(const S: string): boolean;
var i, n : integer;
begin
  n:=0;
  for i:=1 to Length(S) do begin
    case S[i] of
      '(': n:=n+1;
      ')': n:=n-1;
    end;
    if n<0 then break;  { ошибка }
  end;
  Check:= n=0;
end;

var F: Text;
    S: string;

begin
  Assign(F,'Test.in');
  Reset(F);
  while not Eof(F) do begin
    Readln(F, S);
    if not Check(S) then Writeln(S);
  end;
  Writeln('OK'); Readln;
end.
```

Г) Дребезг контактов – с этим явлением борются специалисты по электронике. Дребезг возникает при замыкании-размыкании кнопок, тумблеров, реле и других подобных устройств. Сигнал от контактов поступает в микропроцессор с некоторой периодичностью, скажем, 100 раз в секунду. Если контакт разомкнут, микропроцессор принимает «0», а если замкнут – «1». В ходе замыкания-размыкания контакт неустойчив, и процессор получает несколько чередующихся нулей и единиц, – его программа должна отфильтровать эти ложные срабатывания.

Ваша программа будет моделировать поведение микропроцессора. Входной файл содержит последовательность нулей и единиц (по одному символу в строке). Из первой строки берется исходное значение сигнала, а дальше сигнал на выходе программы

формируется так: если три подряд идущие значения совпадают, то берется это новое значение, а иначе сохраняется текущее, например.

На входе	На выходе
0	0
1	0
0	0
1	0
1	0
1	1
0	1

Выходной файл должен содержать две колонки: входной и выходной сигналы.

```
var F_in, F_out: Text;
    C: char;              { входной сигнал = 0/1 }
    C1, C2, C3 : char;    { три последних значения }
    Res : char;           { выходной сигнал }
    cnt : integer;        { вспомогательный счетчик до трех }

begin
  Assign(F_in,'Test.in');
  Assign(F_out,'Test.out');
  Reset(F_in);
  Rewrite(F_out);
  { читаем исходное значение сигнала }
  if not Eof(F_in) then begin
    Readln(F_in, C);
    C1:=C; C2:=C; C3:=C; Res:=C;
    Writeln(F_out, C,' ',Res);
  end;
  { обработка последующих строк входного файла }
  cnt:=0;
  while not Eof(F_in) do begin
    Readln(F_in, C);
    { циклически обновляем самое старое значение сигнала }
    case cnt of
      0: C1:= C;
      1: C2:= C;
      2: C3:= C;
    end;
    cnt:= (cnt+1) mod 3;  { 0,1,2,0,1...}
    { если все три совпадают, обновляем выходной сигнал }
    if (C1=C2) and (C1=C3) then Res:=C;
    Writeln(F_out, C,' ', Res);
  end;
  Close(F_in); Close(F_out);
  Writeln('OK'); Readln;
end.
```

Глава 29

А) Напишите программу для преобразования второго варианта базы данных «Police.txt» (с несколькими числами в строке) в первый вариант (по одному числу в строке). Или слабо?

```
var F1, F2: text;
    N: integer;

begin
  Assign(F1, 'Police.in');  Reset(F1);
  Assign(F2, 'Police.out'); Rewrite(F2);

  while not Eof(F1) do begin
    Read(F1, N);
    Writeln(F2,N);
  end;
  Close(F1);  Close(F2);
end.
```

Б) Можно ли при решении предыдущей задачи назначить одно и то же имя как входному, так и выходному файлам? Испытайте свое решение.

```
Нельзя
```

Глава 30

А) Функция **Trunc** выделяет целую часть вещественного числа, например.

```
Writeln (Trunc( 12.345 ));    { 12 }
```

Исследуйте её и придумайте способ выделения дробной части вещественного числа. Напишите подходящую функцию и программу для её проверки.

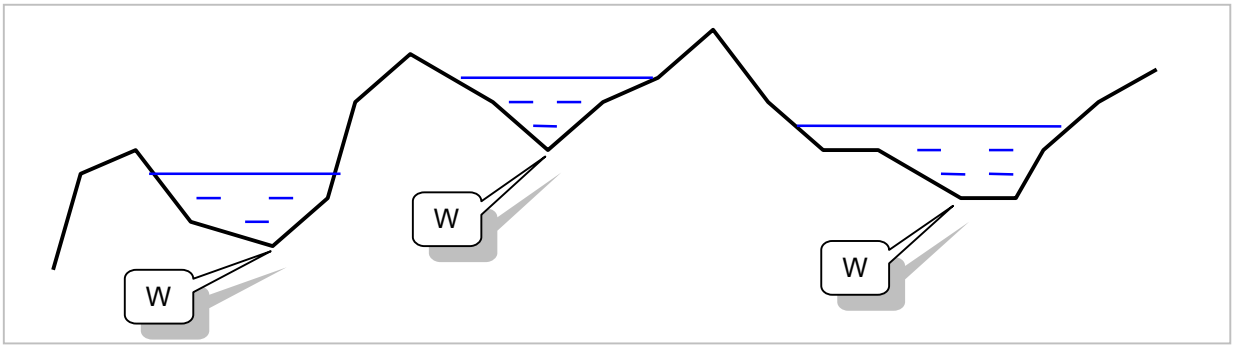
```
var R: real;

function Fractal(arg : real): real;
begin
  Fractal:= arg - Trunc(arg);
end;

begin
  repeat
    Write('Введите R=');
    Readln(R);
    Writeln(Fractal(R):10:5);
  until R<0;
end.
```

В) Сформируйте файл «Numbers.txt», поместив в него 100 случайных чисел в диапазоне от 0 до 999 (некоторые числа могут повторяться). Затем найдите в этом файле: 1) максимальное и минимальное число; 2) сумму всех чисел; 3) среднее арифметическое – напечатайте его с точностью 2 знака после точки.

```
const CFileName = 'Numbers.txt';
      CNumbers = 100;

procedure CreateDataFile;       { создание файла с числами }
var F: Text;
    i, N : integer;
begin
  Assign(F, CFileName);
  Rewrite(F);
  for i:=1 to CNumbers do begin
    Write(F, Random(500)+Random(500):5);
    if i mod 10 = 0 then Writeln(F);
  end;
  Close(F);
end;

procedure Handle;    { обработка файла с числами }
var F: Text;
    N, min, max, Sum, Cnt : integer;
begin
  min:=1000; { минимум }
  max:=0;    { максимум }
  Sum:=0;    { сумма }
  Cnt:=0;    { счетчик }
  Assign(F, CFileName);
```

```
  Reset(F);
  while not Eof(F) do begin
    if Eoln(F) then Readln(F); { пропуск конца строки }
    Read(F, N);
    if N<min then min:=N;
    if N>max then max:=N;
    Sum:= Sum+N;
    Cnt:= Cnt+1;
    if Cnt=CNumbers then break;
  end;
  Close(F);
  Writeln('Минимальное= ', min);
  Writeln('Максимальное= ', max);
  Writeln('Сумма= ', Sum);
  Writeln('Среднее= ', Sum/Cnt : 12:2);
end;

begin
  CreateDataFile;
  Handle;
  Writeln('ok'); Readln
end.
```

Г) Сканирование марсианской поверхности дало файл, содержащий высоту отдельных его точек вдоль одного из направлений, – пусть это будет файл «Numbers.txt» из предыдущей задачи. Найдите точки, где вероятней всего обнаружить марсианскую воду, – на следующем ниже рисунке они обозначены буквами W. Программа должна напечатать две колонки: порядковый номер точки относительно начала файла (счет от нуля) и высоту точки (такие точки математики называют локальными минимумами).



Рельеф марсианской поверхности

```
const CFileName = 'Numbers.txt';

var F : Text;
    h1, h2, h3 : integer; { высоты для трех соседних точек }
    h : integer;          { очередная точка }
    N : integer;          { порядковый номер точки }
begin
  Assign(F, CFileName); Reset(F);
  { две первых читаем из файла }
  if not eof(F) then Read(F, h1);
  if not eof(F) then Read(F, h2);
  N:=0;
  while not eof(F) do begin
    if Eoln(F) then Readln(F);  { пропуск конца строки }
    Read(F, h);
    N:=N+1;
    if h3<>h then begin  { если две соседних не совпадают }
      h3:=h;
      if (h2<h1) and (h2<h3) then Writeln(N:5, h2:10);
      h1:=h2; h2:=h3  { сдвиг точек }
    end;
  end;
  Write('OK'); Readln;
end.
```

Глава 31

А) Напишите программу для преобразования первого варианта базы данных «Police.txt» (которая содержит по одному числу в строке) во второй вариант (будет содержать по три числа в строке).

```
var N, K: integer;
    F1, F2: Text;
begin
  Assign(F1,'Police.in'); Reset(F1);
  Assign(F2,'Police.out'); Rewrite(F2);
  K:=0;
  while not Eof(F1) do begin
    Read(F1, N);
    Write(F2, N, ' ');
    K:= (K+1) mod 3;  { K= 0, 1, 2, 0, 1, 2 ...}
    if K=0 then Writeln(F2);
  end;
  Close(F1); Close(F2);
end.
```

Глава 32

Б) Перечислимые типы и диапазоны строятся на базе других типов данных (**Byte**, **ShortInt** и так далее). Какие типы данных, по вашему мнению, будут положены в основу следующих диапазонов:

```
var   N : -10..10;          { byte }
      M : -200..200;        { integer }
      R : 0..40000;         { word }
      L : 0..400000;        { longint }
      S : '0'..'9';         { char }
```

В) Процедура печати **Writeln** не способна распечатать название месяца, представленного в перечислении. Придумайте для этого свою процедуру (воспользуйтесь оператором **CASE**).

```
type  TMonth = (Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec);

procedure WriteMonth(arg: TMonth);
begin
  case arg of
    Jan: Writeln('Январь');
    Feb: Writeln('Февраль');
    Mar: Writeln('Март');
    Apr: Writeln('Апрель');
    May: Writeln('Май');
    Jun: Writeln('Июнь');
    Jul: Writeln('Июль');
    Aug: Writeln('Август');
    Sep: Writeln('Сентябрь');
    Oct: Writeln('Октябрь');
    Nov: Writeln('Ноябрь');
    Dec: Writeln('Декабрь');
  end;
end;

var M : TMonth;
begin
  M:= Apr;
  WriteMonth(M);
  M:= Oct;
  WriteMonth(M);
  Readln;
end.
```

Г) «Не думай о секундах свысока…». Штирлицу подарили секундомер, показывающий число секунд с начала суток. Пусть ваша программа переведет это число в привычные часы, минуты и секунды. Подсказки: во-первых, воспользуйтесь операциями **DIV** и **MOD**. Во-вторых, объявляя переменную для секунд, примените вместо типа **INTEGER** тип **LONGINT**, поскольку количество секунд в сутках (86400) не поместится в типе **INTEGER**.

```
var Time : Longint;              { время в секундах }
    Hour, Min, Sec : integer;    { часы, минуты, секунды }

begin
  Write('Введите время в секундах: '); Readln(Time);
  Sec:= Time mod 60;              { секунды }
  Min:= (Time div 60) mod 60;     { минуты }
  Hour:= (Time div 60) div 60;    { часы }
  Write('Время: ',Hour,':',Min,':',Sec);
  Readln;
end.
```

Глава 33

А) Напишите две функции, округляющие вещественное число:

- до большего значения (например: 3.1 → 4; 3.9 → 4);
- до меньшего значения (например: 3.1 → 3; 3.9 → 3).

```
function RoundUp(arg: Extended): Extended;
begin
  RoundUp:= Round(arg+0.5);
end;

function RoundDown(arg: Extended): Extended;
begin
  RoundDown:= Round(arg-0.5);
end;

begin
  Writeln(RoundUp(3.9));
  Writeln(RoundUp(3.1));
  Writeln(RoundUp(-3.9));
  Writeln(RoundUp(-3.1));
  Writeln(RoundDown(3.9));
  Writeln(RoundDown(3.1));
  Writeln(RoundDown(-3.9));
  Writeln(RoundDown(-3.1));
  Readln;
end.
```

Б) Ваша процедура принимает строковую переменную, вычисляет среднее арифметическое кодов её символов и печатает его с двумя цифрами после точки.

```
function CalcStr(const arg: string): Extended;
var i : integer;
    sum : Extended;
begin
  sum:=0;
  for i := 1 to Length(arg) do sum:= sum+Ord(arg[i]);
  CalcStr:= sum / Length(arg);
end;

begin
  Writeln(CalcStr('=abcdef='):10:2);
  Writeln(CalcStr('-123456789-'):10:2);
  Readln;
end.
```

В) Напечатайте с тремя знаками после точки 20 случайных чисел в диапазоне от 0 до 10. Подсказка: для формирования чисел можно делить случайное целое число на другое число, например, **`Random(10000)/1000`**.

```
var i: integer;

begin
  for i:=1 to 20 do Writeln(Random(10000)/1000 :20:3);
  Readln;
end.
```

Г) Напечатайте с тремя знаками после точки несколько случайных чисел в диапазоне от 0 до 10 так, чтобы числа следовали по возрастанию. Подсказка: сравнивайте очередное число с предыдущим.

```
var i: integer;
    N1, N2: extended;

begin
  Randomize;
  N1:=0;
  for i:=1 to 20 do begin
    repeat
      N2:= Random(500*i)/1000;
    until N2>N1;
    N1:=N2;
    Writeln(N2 :20:3);
  end;
  Readln;
end.
```

Д) Напишите программу для подсчета стоимости покупок. Для каждой покупки пользователь вводит два действительных числа: вес покупки (кг) и цену за 1 кг (руб). Признак завершения ввода данных – нулевой вес. Программа должна напечатать общую стоимость с точностью до копейки (два знака после точки) с округлением в большую сторону. Проверьте результат на калькуляторе.

```
var
     Weight, Cost, Sum : extended; { вес, цена, сумма }

begin
  Sum:=0;
  Repeat
    Write('Введите через пробел вес и цену: ');
    Readln( Weight, Cost );
    Sum:= Sum + Weight * Cost;
  Until Weight=0;
  Sum:= Round((100*Sum)+0.5) / 100;   { округление копеек }
  Write('Sum= ', Sum:20:2);
  Readln;
end.
```

E) Квадратный корень. Квадрат – это равносторонний прямоугольник, его площадь вычисляется по формуле **S=D•D**, где **D** – сторона квадрата. А когда площадь **S** известна, и надо определить сторону **D**? Тогда из **S** извлекают квадратный корень (обозначается символом **√)** . Так, если **S=9**, то **D=√9=3**.

Для извлечения корня в Паскале есть функция **SQRT**. Но вы напишите собственную, прибегнув к методу последовательных приближений. В грубом, нулевом приближении примем $D_0=1$. Последующее, более точные значения **D** будем вычислять по формуле

$$D_{i+1} = (D_i + S/D_i)/2$$

Так, при **S=9** получим $D_1=(1+9/1)/2=\underline{5}$, $D_2=(5+9/5)/2=\underline{3.4}$ и так далее, пока абсолютная разность между двумя последовательными значениями **D** станет пренебрежимо мала. Напишите вещественную функцию **MySQRT**, принимающую число, из которого надо извлечь корень, и вычисляющую этот корень с точностью **0.0001**. Внутри процедуры напечатайте промежуточные значения **D**. Подсказка: для D_i и D_{i+1} вам потребуются лишь две локальные переменные.

```
function MySQRT(arg : extended): extended;
var D0, D1 : extended;
begin
  D1:= 1;
  repeat
    D0:=D1;
    D1:= (D0 + arg/D0)/2;
    Writeln(D0:10:4, D1:10:4); { для наблюдения за вычислением }
  until Abs(D1-D0) < 0.0001;
  MySQRT:= D1;
end;

begin
  Writeln(MySQRT(9) :15:4);
  Writeln(MySQRT(25):15:4);
  Writeln(MySQRT(90):15:4);
  Readln;
end.
```

Ж) В тесто кладут четырех главных ингредиента: муку, сахар, яичный порошок и молоко. Все это смешивается в пропорции, заданной рецептом. Например, рецепт **100:5:7:500** означает, что на 100 граммов муки кладут 5 граммов сахара, 7 граммов яичного порошка и 500 граммов молока. У пекаря есть некоторое количество всех ингредиентов, и он хочет замесить из них максимально возможное количество теста, соблюдая рецепт. Ваша программа должна ввести:

- Рецепт – это 4 целых числа.
- Исходное количество ингредиентов – это 4 действительных числа.

Программа должна напечатать:

- Общее количество полученного теста с точностью два знака после точки.
- Остатки ингредиентов – 4 числа с точностью два знака после точки.

```
type TReal = Extended;

var R1, R2, R3, R4 : integer; { Рецепт }
    M1, M2, M3, M4 : TReal;   { Количество ингредиентов }
    P1, P2, P3, P4 : TReal;   { Количество в порциях }
    MinPortion : TReal;       { Минимальная порция }
    Q : TReal;                { Количество теста }
    Q1, Q2, Q3, Q4 : TReal;   { Остатки ингредиентов }

function MinReal(arg1, arg2 : TReal): TReal;
begin
  if arg1<arg2 then MinReal:= arg1 else MinReal:= arg2
end;

begin
  Write('Введите четыре числа для рецепта: ');
  Readln(R1, R2, R3, R4);
  Write('Введите четыре числа - количество ингредиентов: ');
  Readln(M1, M2, M3, M4);
  { Количество в порциях }
  P1:= M1/R1; P2:= M2/R2; P3:= M3/R3; P4:= M4/R4;
  { Минимальная порция }
  MinPortion:= MinReal(MinReal(P1,P2), MinReal(P3,P4));
  Q:= (R1+R2+R3+R4)*MinPortion; { Количество продукта }
  { Остатки ингредиентов }
  Q1:= M1 - R1*MinPortion;
  Q2:= M2 - R2*MinPortion;
  Q3:= M3 - R3*MinPortion;
  Q4:= M4 - R4*MinPortion;
  Writeln('Количество теста: ', Q:10:2);
  Writeln('Остатки ингредиентов: ', Q1:10:2, Q2:10:2,
                                    Q3:10:2, Q4:10:2);
  Readln;
end.
```

Глава 34

А) Найдите две ошибки в следующей программе.

```
var X : TNum;                { Тип TNum в этом месте не определен }
type TNum = integer;
const A = 10;
begin
     X:= A+B;                { переменная B не объявлена }
end.
```

Б) Напишите булеву функцию **Test** и программу для её демонстрации. Функция должна проверять, делится ли без остатка первое число на второе,

```
function Test(N1, N2: integer): boolean;
begin
  Test:= false;
  if (N1 mod N2)=0 then Test:=true;
end;

begin
  Writeln(Test(20, 3)); { false }
  Writeln(Test(20, 4)); { true }
  Readln;
end.
```

В) Напишите целочисленную функцию **Division** для деления первого числа на второе без применения операции **DIV**. Вот примеры вызовов:

```
Writeln( Division(20, 4) );        { 5 }
Writeln( Division (21, 5) );       { 4 }
```

Подсказка: внутри функции вычитайте второе число из первого. Предотвратите деление на ноль (как результат возвращайте ноль). Сделайте два варианта: 1) деление положительных чисел, 2) деление чисел с учетом знака.

```
{ Деление положительных чисел }

function Division(N1, N2: integer): integer;
var R: integer;
begin
  R:=0;
  if N2>0 then
    while N1>=N2 do begin
      N1:= N1-N2;
      Inc(R);
    end;
  Division:= R;
end;
```

```
begin
  Writeln(Division(0,0));
  Writeln(Division(10,0));
  Writeln(Division(0,10));
  Writeln(Division(7,3));
  Writeln(Division(3,7));
  Writeln(Division(20,2));
end.

{ Деление чисел со знаком }

function Division(N1, N2: integer): integer;
var R: integer;
    t1, t2 : integer;
begin
  R:=0;
  t1:= Abs(N1); t2:= Abs(N2);
  if t2>0 then
    while t1>=t2 do begin
      t1:= t1-t2;
      Inc(R);
    end;
  if (N1<0) and (N2>0) or (N1>0) and (N2<0) then R:=-R;
  Division:= R;
end;

begin
  Writeln(Division(10,1));
  Writeln(Division(-10,2));
  Writeln(Division(9,-3));
  Writeln(Division(-12,-5));
end.
```

Г) Пусть ваша программа распечатает все множители (кроме единицы) введенного пользователем целого положительного числа, например:

```
Введите число: 60
2  2  3  5
```

```
procedure Disassemble(arg: integer);
var i: integer;
begin
  i:=2;
  while arg>1 do
    if (arg mod i)=0 then begin
      Write(i:4);
      arg:= arg div i;
    end else begin
      Inc(i);
    end;
  Writeln;
end;

var N: integer;

begin
  repeat
    Write('N= '); Readln(N);
    Disassemble(N)
  until N=0
end.
```

Д) Напишите функцию для ввода целого числа. Она принимает строку-приглашение и возвращает введенное число, например:

```
      X:= GetNumber('Введите стоимость покупки=');
```

```
function GetNumber(arg: string): integer;
var N: Integer;
begin
  Write(arg); Readln(N);
  GetNumber:= N;
end;

begin
  Writeln('Сумма A+B =', GetNumber('A= ') + GetNumber('B= '));
  Readln;
end.
```

Глава 35

А) Полицейская база данных некоторого государства содержит номера всех автомобилей, сгруппированные в ряд множеств. Три множества составлены по типам автомобилей: легковые, грузовые, автобусы. Шесть множеств образованы по цвету автомобилей: множество белых, черных, желтых, красных, синих и зеленых.

- Пересекается ли множество легковых автомобилей с множеством грузовых? А множество желтых автомобилей с множеством черных?

```
Нет
```

- Может ли быть непустым пересечение множества желтых автомобилей с множеством автобусов?

```
Да
```

- Свидетель дорожно-транспортного происшествия сообщил, что с места преступления скрылся **грузовой** автомобиль **синего** цвета. Как можно «вычислить» группу подозреваемых автомобилей?

```
Вычислить пересечение множества грузовых с множеством синих
```

- На улице висит знак: грузовым проезд запрещен. Как определить множество автомобилей, въезд которым разрешен?

```
Сложить множество легковых с множеством автобусов
```

Глава 36

А) Найдите ошибки в следующих операторах.

```
type TNumbers = set of 1..300; { можность множества больше 255 }
     TChars = set of char;
     TBytes = set of byte;

var c1, c2 : TChars;
     b1, b2 : TBytes;
begin
     c1:= [1..9];                { несоответствие типов }
     c2:= ['1'..'9'];
     c2:= c2 + '0';              { несоответствие типов }
     c2:= c2 + [0];              { несоответствие типов }
     b1:= c1;                    { несоответствие типов }
     b2:= b1 + [1,7,3];
     Writeln(b1=b2);
     Writeln(1 in b2);
     Writeln([1] in b2);         { слева должно быть число }
     Writeln(b1 in b2);          { слева должно быть число }
end.
```

Глава 37

А) Напишите процедуры для ввода и вывода множества символов. Может ли здесь счетчиком цикла быть символьная переменная?

```
См. главу 38
```

В) На основе первого варианта директорской программы придумайте способ поиска учеников, записавшихся более чем в один кружок. Или слабо?

```
R = S1*S2 + S1*S3 + S2*S3;
```

Г) Напишите две функции, принимающие строку и возвращающие:

- строку, в которой символы исходной строки встречаются лишь по разу и следуют в алфавитном порядке, например «PASCAL» → «ACLPS»;
- то же, но порядок следования символов такой же, как в исходной строке, например «PASCAL» → «PASCL».

```
function FilterABC(const arg: string): string;
var Res: string;
    S : set of char;
    i : integer;
    c: Char;
begin
  S:=[];
  for i:= 1 to Length(arg) do S:= S + [ arg[i] ];
  Res:='';
  for c:= Char(0) to Char(255) do
    if c in S then Res:= Res + c;
  FilterABC:= Res;
end;

function Filter(const arg: string): string;
var Res: string;
    S : set of char;
    i : integer;
begin
  S:=[];
  for i:= 1 to Length(arg) do S:= S + [ arg[i] ];
  Res:='';
  for i:=1 to Length(arg) do
    if arg[i] in S then begin
      Res:= Res + arg[i];
      S:= S - [ arg[i] ];
    end;
  Filter:= Res;
end;

begin
  Writeln('По алфавиту: ', FilterABC('PASCAL') );
  Writeln('В исходном порядке: ', Filter('PASCAL') );
  Writeln('OK'); Readln;
end.
```

Глава 38

Придумайте, как выявить тех, кто состоит:

- в трех кружках:

```
R:= S1*S2*S3;
```

- в двух кружках и не более;

```
R:= S1*S2 + S2*S3 + S1*S3 - S1*S2*S3;
```

- только в одном из кружков.

```
R:= (S1+S2+S3) - (S1*S2 + S2*S3 + S1*S3);
```

Б) В небольшом островном государстве действовали забавные законы относительно транспортных средств – автобусов, грузовиков, легковушек.

Во-первых, общее количество автомобилей на острове не должно было превышать 256. Автомобилям назначались номера с 0 до 255, при этом соблюдались следующие правила.

Номера, делившиеся без остатка на 7, назначались автобусам. Те, что делились без остатка на 5, назначались грузовикам, а все прочие – легковушкам. Так, например, номера 35 и 70 доставались автобусам, а не грузовикам, хоть и делились на 5.

Похожие правила применялись и к окраске автомобилей, а именно.

Если номер авто делился на 4, его красили красным, если на 3 – желтым, если на 2 – белым, а остальные автомобили красили черным.

- Сформируйте три множества по классам автомобилей – автобусы, грузовики и легковушки. Вычислите количество машин каждого класса (Ответ: 37, 44, 175).
- Сформируйте четыре множества по цвету автомобилей – красные, желтые, белые и черные. Определите количество машин каждого цвета (Ответ: 64, 64, 43, 85).
- Столица того государства – деревня Кокосовка – страдала от пробок. Для борьбы с ними ввели ограничение на въезд транспорта. Так, в один из дней недели в столицу пускали только красные легковушки, белые грузовики и все автобусы. Найдите номера всех этих машин. Сколько всего автомобилей могло въехать в столицу в тот день?

```
type TAuto = set of byte;

procedure WriteSet(const aSet : TAuto);
var k : integer;
begin
  for k:=0 to 255 do if k in aSet then Write(k:4);
  Writeln;  Writeln('---');
end;
```

```
function Count(const aSet : TAuto): integer;
var k, r : integer;
begin
  r:=0;
  for k:=0 to 255 do if k in aSet then Inc(r);
  Result:=r;
end;

var  Bus, Truck, Car : TAuto;
     Red, Yellow, White, Black : TAuto;

    i: integer;
    R: TAuto;

begin
  Bus:=[]; Truck:=[]; Car:=[];
  Red:=[]; Yellow:=[]; White:=[]; Black:=[];

  for i := 0 to 255 do begin
    if (i mod 7 = 0) then Bus:= Bus+[i]
      else
    if (i mod 5 = 0) then Truck:= Truck+[i]
      else
    Car:= Car+[i];
  end;

  for i := 0 to 255 do begin
    if (i mod 4 = 0) then Red:= Red+[i]
      else
    if (i mod 3 = 0) then Yellow:= Yellow+[i]
      else
    if (i mod 2 = 0) then White:= White+[i]
      else
    Black:= Black+[i];
  end;

  Writeln(Count(Bus));   WriteSet(Bus);
  Writeln(Count(Truck)); WriteSet(Truck);
  Writeln(Count(Car));   WriteSet(Car);
  Readln;

  Writeln(Count(Red));   WriteSet(Red);
  Writeln(Count(Yellow));WriteSet(Yellow);
  Writeln(Count(White)); WriteSet(White);
  Writeln(Count(Black)); WriteSet(Black);
  Readln;

  R:= Red*Car + White*Truck + Bus;
  Writeln(Count(R)); WriteSet(R);
  Readln;
end.
```

Г) Генерация пароля длиной не менее восьми символов. В пароль входят символы трёх сортов: цифры, маленькие и большие латинские буквы, например: «**7UpJ7rsT**», «**PasCal701**». Сделайте четыре варианта так, чтобы соблюдались следующие условия:

- символ каждого сорта входит в пароль не менее двух раз, некоторые символы могут повторяться;
- все символы пароля уникальны (примените множество);
- символы одного сорта не соседствуют, например: «**Pa7sCaL5**», уникальность символов не требуется;
- символы одного сорта не соседствуют и все символы уникальны.

```
{ 1. символ каждого сорта входит в пароль не менее двух раз,
     некоторые символы могут повторяться }

function PassWord_1: string;
var S: string;  { результат }
    v: integer; { вариант очередного символа }
    n1,n2,n3: integer;  { счетчики }
begin
  S:='';
  n1:=0; n2:=0; n3:=0;
  while (Length(S)<8) or (n1<2) or (n2<2) or (n3<2) do begin
    v:= Random(3);
    case v of
      0: begin S:=S + Char(Random(10)+Ord('0')); Inc(n1) end;
      1: begin S:=S + Char(Random(26)+Ord('a')); Inc(n2) end;
      2: begin S:=S + Char(Random(26)+Ord('A')); Inc(n3) end;
    end;
  end;
  PassWord_1:=S;
end;

{ 2. все символы уникальны }

function PassWord_2: string;
var S: string;  { результат }
    v: integer; { вариант очередного символа }
    C: char;    { очередной символ }
    T: set of char;
begin
  S:=''; T:=[ ];
  while Length(S)<8+Random(5) do begin
    v:= Random(3);
    case v of
      0: C:= Char(Random(10)+Ord('0'));
      1: C:= Char(Random(26)+Ord('a'));
      2: C:= Char(Random(26)+Ord('A'));
    end;
    if not (C in T) then begin
      T:= T+[C];
      S:= S+C;
    end;
  end;
```

```
  PassWord_2:=S;
end;

{ 3. сорта чередуются, символы не уникальны }

function PassWord_3: string;
var S: string;  { результат }
    v: integer; { вариант очередного символа }
    C: char;    { очередной символ }
begin
  S:='';
  v:= Random(3);
  while Length(S)<8+Random(5) do begin
    case v of
      0: C:= Char(Random(10)+Ord('0'));
      1: C:= Char(Random(26)+Ord('a'));
      2: C:= Char(Random(26)+Ord('A'));
    end;
    S:= S+C;
    v:= (v+1+Random(2)) mod 3;
  end;
  PassWord_3:=S;
end;

{ 4. сорта чередуются, символы уникальные }

function PassWord_4: string;
var S: string;  { результат }
    v: integer; { вариант очередного символа }
    C: char;    { очередной символ }
    T: set of char;
begin
  S:=''; T:=[ ];
  v:= Random(3);
  while Length(S)<8+Random(5) do begin
    case v of
      0: repeat
           C:= Char(Random(10)+Ord('0'));
           if not (C in T) then begin T:= T+[C]; Break end
         until false;
      1: repeat
           C:= Char(Random(26)+Ord('a'));
           if not (C in T) then begin T:= T+[C]; Break end
         until false;
      2: repeat
           C:= Char(Random(26)+Ord('A'));
           if not (C in T) then begin T:= T+[C]; Break end
         until false;
    end;
    S:= S+C;
    v:= (v+1+Random(2)) mod 3;
  end;
  PassWord_4:=S;
end;


var F: text;
    i: integer;
```

```
begin
  Randomize;
  Assign(F, 'a_38_g.out'); Rewrite(F);
  for i := 1 to 10 do Writeln(F, PassWord_1);
  for i := 1 to 10 do Writeln(F, PassWord_2);
  for i := 1 to 10 do Writeln(F, PassWord_3);
  for i := 1 to 10 do Writeln(F, PassWord_4);
  Close(F);
end.
```

Файл «a_38_g.out»

```
V3d3b1MU
2e45XFJl
4FxPK52Xx
EtNvWeE75
31U5mJ142UWYc
P1x14uuyB
mIPnfl48
qUet8w9Z
8V4iDLbE
QjL6KY3A4Sm

9lmuvEby2o
DMBYyrUgJq
E94p7c36Nu5
45s2U9JfXF
X51I2gPNm
b4Pn7iruo
NS25gbp78
367UHbTp
PVm7x2oT0N
qA7D4ObK

G4v1qIl1W
oK7jW5Q8
1PnY2wK0q
Ca3yO0A8Vc
2n1lQ7x5m9
5u0K3y8lO
9m6HcIfQ
SsH9My4G
M6s3j2K3Dg3
E5AdI0V6hT

n3cAkYvH9d
Pn8ZeW2O5o
k6CoM3B2q
8A3NoB4t
4wA8m9JpLj
xMaBmD7e2Y
3y6I8bWa5
U9A0O2ZlFq
8jV7cNtU
5D8pL7w2Q3
```

Д) Напишите булевы функции, проверяющие, является ли введенная пользователем строка правильно сформированным паролем согласно условиям предыдущей задачи.

```
{ Вспомогательная функция возвращает группу символов:
  1 - для цифр,
  2 - для малых латинских,
  3 - для больших латинских
  0 - для прочих
}

function Group(arg: char): integer;
begin
  Group:=0;
  if arg in ['0'..'9']
    then Group:=1
  else if arg in ['a'..'z']
    then Group:=2
  else if arg in ['A'..'Z']
    then Group:=3
end;

{ Test_1 возвращает true, если все символы допустимы,
  и символов каждого сорта не менее двух,
  и длина пароля не менее 8 символов }

function Test_1(const arg: string): boolean;
var i: integer;
    g: integer;  { группа (сорт) символа }
    c1, c2, c3 : integer; { три счетчика }
begin
  c1:=0;  c2:=0;  c3:=0;
  for i:=1 to Length(arg) do begin
    g:= Group(arg[i]);
    if g=0 then begin
      Test_1:= false;
      Break;
    end;
    { наращиваем счетчики групп }
    if g=1
      then Inc(c1)
    else if g=2
      then Inc(c2)
    else Inc(c3);
  end;
  Test_1:= (Length(arg)>7) and (c1>1) and (c2>1) and (c3>1);
end;

{ Test_2 возвращает true, если все символы допустимы,
  и все они уникальны,
  и длина пароля не менее 8 символов }

function Test_2(const arg: string): boolean;
var i: integer;
    T: set of char;
begin
  Test_2:= Length(arg)>7;
  T:=[ ];
  for i:=1 to Length(arg) do begin
```

```
    if not (arg[i] in ['0'..'9','a'..'z','A'..'Z']) or (arg[i] in T)
      then begin
        Test_2:= false;
        Break;
      end;
    T:= T + [arg[i]];
  end;
end;

{ Test_3 возвращает true, если все символы допустимы,
  и символы одной группы не соседствуют,
  и длина пароля не менее 8 символов }

function Test_3(const arg: string): boolean;
var i: integer;
    g1, g2 : integer; { сорт символов }
begin
  Test_3:= Length(arg)>7;
  g2:=-1;
  for i:=1 to Length(arg) do begin
    g1:= Group(arg[i]);
    if (g1 = 0) or (g1 = g2) then begin
      Test_3:= false;
      Break;
    end;
    g2:= g1;
  end;
end;

{ Test_4 возвращает true, если все символы допустимы,
  и все они уникальны,
  и символы одной группы не соседствуют,
  и длина пароля не менее 8 символов }

function Test_4(const arg: string): boolean;
var i: integer;
    g1, g2 : integer; { сорт (группа) символов }
    T: set of char;
begin
  Test_4:= Length(arg)>7;
  g2:=-1;
  T:=[ ];
  for i:=1 to Length(arg) do begin
    g1:= Group(arg[i]);
    if (g1 = 0) or (g1 = g2) or (arg[i] in T) then begin
      Test_4:= false;
      Break;
    end;
    g2:= g1;
    T:= T + [arg[i]];
  end;
end;


var F_In, F_Out: Text;
    S: String;
    R: String;
begin
```

```
  Assign(F_In, 'a_38_d.in'); Reset(F_In);
  Assign(F_Out,'a_38_d.out'); Rewrite(F_Out);
  while not Eof(F_In) do begin
    Readln(F_In,S);
    { В строке T формируем результат тестирования
      четырьмя функциями }
    if Test_1(S) then R:='T' else R:='F';
    if Test_2(S) then R:=R+' T' else R:=R+' F';
    if Test_3(S) then R:=R+' T' else R:=R+' F';
    if Test_4(S) then R:=R+' T' else R:=R+' F';
    Writeln(F_Out, S);
    Writeln(F_Out, R);
  end;
  Close(F_In); Close(F_Out);
end.
```

Файл «a_38_в.out» (в качестве входного файла «a_38_d.in» взят файл «a_38_g.out»)

```
V3d3b1MU
T F F F
2e45XFJl
T T F F
4FxPK52Xx
T F F F
EtNvWeE75
T F F F
31U5mJ142UWYc
T F F F
P1x14uuyB
T F F F
mIPnfl48
T T F F
qUet8w9Z
T T F F
8V4iDLbE
T T F F
QjL6KY3A4Sm
T T F F
9lmuvEby2o
F T F F
DMBYyrUgJq
F T F F
E94p7c36Nu5
T T F F
45s2U9JfXF
T T F F
X51I2gPNm
T T F F
b4Pn7iruo
F T F F
NS25gbp78
T T F F
367UHbTp
T T F F
PVm7x2oT0N
T T F F
qA7D4ObK
T T T T
```

```
G4v1qIl1W
T F T F
oK7jW5Q8
T T T T
1PnY2wK0q
T T T T
Ca3yO0A8Vc
T T T T
2n1lQ7x5m9
F T T T
5u0K3y8lO
T T T T
9m6HcIfQ
T T T T
SsH9My4G
T T T T
M6s3j2K3Dg3
T F T F
E5AdI0V6hT
T T T T
n3cAkYvH9d
T T T T
Pn8ZeW2O5o
T T T T
k6CoM3B2q
T T T T
8A3NoB4t
T T T T
4wA8m9JpLj
T T T T
xMaBmD7e2Y
T T T T
3y6I8bWa5
T T T T
U9A0O2ZlFq
T T T T
8jV7cNtU
T T T T
5D8pL7w2Q3
T T T T
```

Глава 39

А) Массив **A** объявлен следующим образом.

```
var   A : array ['a'..'z'] of integer;
      C: char;
```

Допустимо ли такое объявление? Сколько элементов содержит этот массив?

```
Допустимо, массив содержит 26 элементов.
```

Какие из указанных ниже операторов будут (или могут) вызывать ошибки нарушения диапазонов?

```
      A['s']:= 10;                { допустимо }
      A['R']:= 10;                { нарушение диапазона }
      C:='d'; A[C]:= 10;          { допустимо }
      Readln(C); A[C]:= 10;{ может вызвать ошибку исполнения }
```

Глава 40

А) Напишите программу для подсчета различных цифр во входном файле полицейской базы данных (считать надо именно цифры, а не числа!).

```
var Counts : array['0'..'9'] of integer;
    c: char;
    F: Text;
begin
  for c := '0' to '9' do Counts[c]:=0;
  Assign(F, 'Police.txt'); Reset(F);
  while not Eof(F) do begin
    Read(F,c);
    if c in ['0'..'9']
      then Inc(Counts[c]);
  end;
  for c := '0' to '9' do Writeln(c, Counts[c]:7);
  Readln;
end.
```

Б) Объявите массив из сотни целых чисел, заполните его случайными числами в диапазоне от 0 до 255 и распечатайте этот массив.

В) Найдите в массиве (задание Б) все элементы, хранящие число 7 (если таковые найдутся). Напечатайте индексы элементов, которые содержат это число.

```
const CSize = 100;
type TArray = array [1..CSize] of word;
var Arr : TArray;
    i : integer;
begin
  Randomize;
  for i:=1 to CSize do Arr[i]:= Random(256);
  Writeln('Все элементы массива :');
  for i:=1 to CSize do begin
    Write(i:4,'=',Arr[i]:4);
    if i mod 10 = 0 then Writeln;
  end;
  Writeln('Элементы, содержащие число 7 :');
  for i:=1 to CSize do if Arr[i]=7 then Writeln(i);
  Readln;
end.
```

Г) Заполните массив (задание Б) случайными числами в диапазоне от 0 до 255 так, чтобы ни одно из них не повторялось. Воспользуйтесь вспомогательным множеством чисел, где будут запоминаться сгенерированные ранее числа.

```
const CSize = 100;
type TArray = array [1..CSize] of word;
var Arr : TArray;
    S : set of byte;
    i, n : integer;
begin
  Randomize;
  S:=[ ];
  for i:=1 to CSize do begin
    repeat
      n:= Random(256)
    until not (n in S);
    S:= S+ [n];
    Arr[i]:= n;
  end;
  Writeln('Все элементы массива :');
  for i:=1 to CSize do begin
    Write(i:4,'=',Arr[i]:4);
    if i mod 10 = 0 then Writeln;
  end;
  Readln;
end.
```

Д) Найдите в массиве (задание Г) наименьшее и наибольшее числа, напечатайте их, а также соответствующие им индексы элементов массива.

```
const CSize = 100;
type TArray = array [1..CSize] of word;
var Arr : TArray;
    S : set of byte;
    i, n : integer;
    min, max : integer;
    imin, imax : integer;
begin
  Randomize;
  S:=[ ];
  for i:=1 to CSize do begin
    repeat n:= Random(256) until not (n in S);
    S:= S+ [n];
    Arr[i]:= n;
  end;
  Writeln('Все элементы массива :');
  for i:=1 to CSize do begin
    Write(i:4,'=',Arr[i]:4);
    if i mod 10 = 0 then Writeln;
  end;
  min:=Arr[1]; max:=Arr[1];
  for i:=1 to CSize do begin
    if min > Arr[i] then begin
      min:= Arr[i];
      imin:= i;
    end;
```

```
    if max < Arr[i] then begin
      max:= Arr[i];
      imax:= i;
    end;
  end;
  Writeln('imin= ',imin:4, '  min= ',min);
  Writeln('imax= ',imax:4, '  max= ',max);
  Readln;
end.
```

Е) Вращение массива вправо. Объявите массив из 10 чисел и заполните его случайным образом. Напишите процедуру, перемещающую 1-й элемент на 2-е место, 2-й — на 3-е место и т.д. Последний элемент должен занять 1-е место.

Ж) Вращение массива влево. Напишите процедуру для перемещения 2-го элемента на 1-е место, 3-го — на 2-е место и т.д. А первый элемент должен стать последним.

```
const CSize = 10;
type TArray = array [1..CSize] of word;
var Arr, ArrLeft, ArrRight : TArray;

procedure ShowArray(const msg: string; const arg : TArray);
var i : integer;
begin
  Writeln(msg);
  for i:=1 to CSize do Write(arg[i]:4);
  Writeln;
end;

procedure Left(var arg : TArray);
var i, temp : integer;
begin
  temp:= arg[1];
  for i:=1 to CSize-1 do arg[i]:= arg[i+1];
  arg[CSize]:= temp;
end;

procedure Right(var arg : TArray);
var i, temp : integer;
begin
  temp:= arg[CSize];
  for i:= CSize downto 2 do arg[i]:= arg[i-1];
  arg[1]:= temp;
end;

var i : integer;
begin
  Randomize;
  for i:=1 to CSize do Arr[i]:= Random(256);
  ShowArray('Исходный массив: ',Arr);
  ArrLeft:= Arr;  Left(ArrLeft);
  ShowArray('Сдвинутый по кругу влево: ',ArrLeft);
  ArrRight:= Arr;  Right(ArrRight);
  ShowArray('Сдвинутый по кругу вправо: ',ArrRight);
  Readln;
end.
```

И) Напишите функцию для подсчета количества номеров в полицейской БД при условии, что одна строка может содержать несколько номеров, а некоторые строки (в т.ч. в конце файла) могут быть пустыми.

```
function CalcNumbers(var aFile: text): integer;
var cnt,n: integer;
begin
  cnt:=0;
  while not Eof(aFile) do begin
    { Пропуск пустых строк }
    while Eoln(aFile) do
      if Eof(aFile) then Break else Readln(aFile);
    if Eof(aFile) then Break;
    Read(aFile, n);
    Inc(cnt);
  end;
  CalcNumbers:= cnt;
end;

var F: Text;

begin
  Assign(F, 'Test.txt'); Reset(F);
  Writeln(CalcNumbers(F));
  Close(F);
  Readln;
end.
```

Глава 41

А) Напишите программу для сортировки фамилий учеников в алфавитном порядке. Программа должна сортировать как по возрастанию, так и по убыванию фамилий (на выбор пользователя).

```
const CSize = 10; { размер массива }

type
  TItem = string; { тип для фамилий }
  TFams = array [1..CSize] of TItem;

var  Fams : TFams;  { массив фамилий }

  { функция сравнения фамилий }

function Compare(Item1, Item2 : TItem; Direct: boolean): boolean;
begin
  if Direct
    then Compare:= Item1 > Item2
    else Compare:= Item1 < Item2
end;

{ Процедура "пузырьковой" сортировки, Direct - направление сорт-ки }

procedure BubbleSort(var arg: TFams; Direct: boolean);
var  i, j : integer;
     t: TItem;
begin
  for i:= 1 to CSize-1 do    { внешний цикл }
    for j:= 1 to CSize-i do  { внутренний цикл }
      if Compare(arg[j], arg[j+1], Direct) then
      begin
        t:= arg[j];          { временно запоминаем }
        arg[j]:= arg[j+1];   { следующий -> в текущий }
        arg[j+1]:= t;        { текущий -> в следующий }
      end;
end;

var i: integer;
    F: Text;

begin {--- Главная программа ---}
  { ввод фамилий из файла }
  Assign(F, 'Fams.txt'); Reset(F);
  i:=1;
  while not Eof(F) and (i<=CSize) do begin
    Readln(F, Fams[i]);
    Inc(i);
  end;
  Close(F);
  Writeln('До сортировки:');
  for i:=1 to CSize do Writeln(Fams[i]);
  Readln;
  Writeln('По убыванию:');
  BubbleSort(Fams, false);
  for i:=1 to CSize do Writeln(Fams[i]:3);
```

```
  Readln;
  Writeln('По возрастанию:');
  BubbleSort(Fams, true);
  for i:=1 to CSize do Writeln(Fams[i]:3);
  Readln;
end.
```

Г) Напишите функцию, проверяющую, упорядочен ли числовой массив (функция должна вернуть **TRUE**, если массив упорядочен по возрастанию). Массив внутрь функции передайте параметром по ссылке.

```
const CSize = 10;
type TArray = array [1..CSize] of word;
var Arr1, Arr2 : TArray;

function IsOrdered(const arg: TArray): boolean;
var i: integer;
begin
  IsOrdered:= true;
  for i:=1 to CSize-1 do
    if arg[i+1] < arg[i] then begin
      IsOrdered:= false;
      Break;
    end;
end;

var i: integer;

begin
  for i:=1 to CSize do Arr1[i]:= Random(256);
  for i:=1 to CSize do Arr2[i]:= i;
  if IsOrdered(Arr1)
    then Writeln('1-й массив упорядочен')
    else Writeln('1-й массив неупорядочен');
  if IsOrdered(Arr2)
    then Writeln('2-й массив упорядочен')
    else Writeln('2-й массив неупорядочен');
  Readln;
end.
```

Глава 42

А). Будет ли линейный поиск работать быстрее в сортированном массиве? Проверьте на практике.

```
Неудачный поиск можно прекращать раньше, и тем самым экономить время.
```

Б) Сколько шагов двоичного поиска потребуется для массива из миллиона элементов? А из миллиарда? Сравните с числом шагов при линейном поиске.

```
Для миллиона – порядка 20 шагов, для миллиарда – порядка 30.
```

Д) Папа Карло опасался Буратино, и прятал спички в сейфе. Код замка из четырех цифр он доверил лишь своему приятелю – честному малому Джузеппе, который не поддавался ни на какие уговоры деревянного мальчишки. Тогда тот пустился на хитрость. Ладно, – предложил Буратино, – не можешь открыть мне код, – не надо. Давай тогда в игру сыграем: я буду задавать вопросы, а ты отвечай только «да» или «нет». Первый вопрос был таким: код замка больше 5000? Через несколько минут Буратино уже рылся в папином сейфе. Сделайте программу для быстрого угадывания числа методом Буратино. Роль Буратино (угадывателя) должен исполнять компьютер.

```
var L, M, R : integer;
    answer: char;
begin
  Write('Запишите на бумаге число от 0 до 9999, затем нажмите Enter');
  Readln;
  L:=0; R:=9999;
  repeat
    M:= (L+R) div 2;
    Write('Ваше число больше ',M,' ? '); Readln(answer);
    if answer='y'
      then L:=M+1
      else R:=M
  until L=R;
  Write('Вы задумали число ', L, ' нажмите Enter');
  Readln;
end.
```

Глава 44

Б) Напишите функцию для приведения любой буквы к верхнему регистру (включая и русские). Подсказка: вспомните о таблице кодировки.

```
const { типизированные константы для русских букв }

  HighChars: string = 'АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЭЮЯ';
  LowChars : string = 'абвгдеёжзийклмнопрстуфхцчшщъыэюя';

function High(arg: char): char;
var i: integer;
begin
  High:= arg;
  if arg in ['a'..'z']
    then High:= Upcase(arg)
    else for i := 1 to Length(LowChars) do
           if arg= LowChars[i] then begin
             High:= HighChars[i];
             break;
           end
end;

function Low(arg: char): char;
var i: integer;
begin
  Low:= arg;
  if arg in ['A'..'Z']
    then Low:= Char( Ord(arg) + Ord('z') - Ord('Z') )
    else for i := 1 to Length(LowChars) do
           if arg= HighChars[i] then begin
             Low:= LowChars[i];
             break;
           end
end;

function HighStr(const arg: string): string;
var i: integer; s: string;
begin
  s:='';
  for i := 1 to Length(arg) do s:= s+ High(arg[i]);
  HighStr:= s;
end;

function LowStr(const arg: string): string;
var i: integer; s: string;
begin
  s:='';
  for i := 1 to Length(arg) do s:= s+ Low(arg[i]);
  LowStr:= s;
end;

begin
  Writeln(HighStr('Pascal Паскаль 123'));
  Writeln( LowStr('Pascal Паскаль 123'));
  Readln;
end.
```

В) Напишите функцию для приведения любой буквы к нижнему регистру.

```
См. предыдущий пример
```

З) Дана строка. Напишите булеву функцию, определяющую, является ли она палиндромом (палиндром – это строка, которая читается справа налево и слева направо одинаково).

```
function IsPalindrom(const arg: string): boolean;
var i: integer;
begin
  IsPalindrom:= true;
  for i:=1 to Length(arg) div 2 do begin
    if arg[i] <> arg[Length(arg)-i+1] then begin
      IsPalindrom:= false;
      Break
    end
  end;
end;

begin
  Writeln(IsPalindrom('ABBA'));
  Writeln(IsPalindrom('Чепуха'));
  Readln;
end.
```

Ж) Строка содержит несколько слов (предложение). Напишите программы для решения следующих задач.

- Напечатать в столбик отдельные слова введённого предложения.
- Определить количество слов в строке.
- Равномерно расставить пробелы между словами так, чтобы длина строки стала равной 80 символам (исходная строка короче 80).

```
const CSize = 40; { максимальное количество слов в строке }
type  TArr = array [1..40] of string;
var   Arr : TArr;        { массив слов }
      StrIn : string;    { исходная строка }
      StrOut : string;   { результирующая строка }
      WordCount : integer;  { количество слов в предложении }

{ Разложение слов по элементам массива }

procedure Parse(const S: string; var A: TArr; var Cnt: integer);
var i: integer;
    w: integer; { счетчик слов }
    c : char;   { предыдущий символ }
begin
  for i:=1 to CSize do A[i]:='';
  w:=0;          { счетчик слов }
  c:= Char(32); { предыдущий символ - пробел }
  for i:=1 to Length(S) do begin
    if (Ord(S[i])>32) then begin
```

```
      if (Ord(c)<=32) then Inc(w);
      A[w]:=A[w] + S[i];
    end;
    c:= S[i];
  end;
  Cnt:= w;  { количество слов в предложении }
end;

{ Формирование строки из отдельных слов массива
  с равномерным распределением пробелов }

function ExpandTo80(const A: TArr): string;
var i, j: integer;
    w: integer;        { количество слов }
    len: integer;      { общая длина всех слов без пробелов }
    Blanks: integer;   { общее кол-во пробелов }
    N: integer;        { кол-во пробелов между соседними словами }
    B: integer;        { B= 0 или 1 - текущая добавка к пробелам }
    Res: string;       { результат }

begin
  { Подсчет общей длины слов и их количества в массиве }
  len:= 0;  i:=1;
  while Length(A[i])>0 do begin
    len:= len + Length(A[i]);
    Inc(w);
    Inc(i);
  end;
  Blanks:= 80-len;     { Общее кол-во пробелов между словами }

  { Начинаем формировать результат }
  B:= 0;  { возможная добавка к пробелам = 0/1 }
  i:=1;  Res:=A[1];   { 1-е слово }
  while w>1 do begin  { пока не все слова обработаны }
    Dec(w);
    N:= Blanks div w; { кол-во пробелов перед следующим словом }
    { Если нацело не делится, то формируем добавку из 0 или 1 пробела
}
    if (Blanks mod w) <> 0 then begin
      N:= N + B;
      B:= (B+1) mod 2;  { B= 0,1,0,1}
    end;
    Blanks:= Blanks - N;  { оставшееся кол-во пробелов }
    { добавляем N пробелов }
    for j:=1 to N do Res:= Res+Char(32);
    { добавляем следующее слово }
    Inc(i);
    Res:= Res + A[i];
  end;
  ExpandTo80:= Res;
end;

{ Распечатка массива слов }

procedure ExpoArray;
var i: integer;
begin
  Writeln('Слов: ',WordCount);
```

```
  for i:=1 to WordCount do Writeln(Arr[i]);
end;

begin
  repeat
    Writeln('Введите строку:');
    Readln(StrIn);
    if Length(StrIn)=0 then Break;
    Parse(StrIn, Arr, WordCount);
    ExpoArray;
    StrOut:= ExpandTo80(Arr);
    Writeln('Конечная строка:');
    Writeln(StrOut);
  until false
end.
```

И) Напишите булеву функцию, определяющую, можно ли из букв первого слова составить второе (например, «клавиша» и «вилка» – **TRUE**). Учитывается только набор букв, а не их количество.

```
type TSet = set of char;

procedure MakeSet(const arg : string; var res: TSet);
var i: integer;
begin
  res:= [ ];
  for i:=1 to Length(arg) do res:= res + [ arg[i] ]
end;

function Test(const arg1, arg2 : string): boolean;
var set1, set2 : TSet;
begin
  MakeSet(arg1, set1);
  MakeSet(arg2, set2);
  Test:= set1 >= set2;
end;

begin
  Writeln(Test('',''));
  Writeln(Test('клавиша','вилка'));
  Writeln(Test('ложка','вилка'));
  Readln;
end.
```

К) Дана строка, содержащая не менее трёх символов. Найти в ней три стоящих подряд символа, у которых сумма кодов максимальна.

```
function FindMax3(const arg: string): string;
var i, j, start, sum, max : integer;
    S: string;
begin
  max:=0;
  start:=1;
  for i:=1 to Length(arg)-2 do begin
    S:= Copy(arg, i, 3);
    sum:=0;
    for j:=1 to Length(S) do sum:= sum + Ord(S[j]);
```

```
    if sum > max then begin
      start:= i;
      max:= sum;
    end
  end;
  FindMax3:= Copy(arg, start, 3);
end;

begin
  Writeln(FindMax3('12345670123'));
  Writeln(FindMax3('990123'));
  Writeln(FindMax3('99'));
  Readln;
end.
```

Л) В строке найти возрастающую последовательность символов наибольшей длины (сравнивайте коды символов).

```
function FindMax(const arg: string): string;
var i : integer;
    start, res  : integer;  { текущее и результир. начало подстроки }
    len, max : integer;     { текущая и максимальная длина }
begin
  max:=0; start:=1; len:=1;
  i:=0;
  while (start+i) < Length(arg) do begin
    if arg[start+i] > arg[start+i+1] then begin
      if len > max then begin
        { запоминаем максимальную длину и начало подстроки }
        max:= len;
        res:= start;
      end;
      start:= start+i+1;
      i:= 0;
      len:=1;
    end else begin
      Inc(i);
      Inc(len);
    end
  end;
  if len > max then begin
    { запоминаем максимальную длину и начало подстроки }
    max:= len;
    res:= start;
  end;
  FindMax:= Copy(arg, res, max);
end;

begin
  Writeln(FindMax('1234123123'));
  Writeln(FindMax('1231230246'));
  Writeln(FindMax('98123456345'));
  Writeln(FindMax('321'));
  Readln;
end.
```

M) Напишите булеву функцию, проверяющую, следуют ли символы строки по неубыванию своих кодов.

```
function IsOrdered(const arg: string): boolean;
var i: integer;
begin
  IsOrdered:= true;
  for i:=1 to Length(arg)-1 do
    if Ord(arg[i+1]) < Ord(arg[i]) then begin
      IsOrdered:= false;
      Break;
    end;
end;

begin
  Writeln(IsOrdered('1248'));
  Writeln(IsOrdered('1240'));
  Writeln(IsOrdered('111223'));
  Readln;
end.
```

H) Напишите функцию для шифрования строки путём перестановки её символов, расположенных на нечётных позициях: первый символ обменивается с последним, третий – с третьим от конца и т.д.

```
procedure Crypt(var arg: string);
var i: integer;
    t: Char;
begin
  for i:=1 to Length(arg) div 2 do begin
    if i mod 2 = 1 then begin
      t:= arg[i];
      arg[i]:= arg[Length(arg)-i+1];
      arg[Length(arg)-i+1]:= t;
    end;
  end;
end;

var S: string;
begin
  S:='procedure Crypt(var arg: string);';
  Crypt(S); { шифрование }
  Writeln(S);
  Crypt(S); { расшифровка }
  Writeln(S);
  Readln;
end.
```

Глава 45

А) Исследуя модель танцевального кружка, можно заметить, что в любой момент одна из двух очередей обязательно пуста. В самом деле, если приходит больше мальчиков, то будет пуста «девчоночья» очередь и наоборот. Можно ли обойтись одной очередью? Придумайте, как это сделать.

Подсказка: добавьте функцию для тестирования очереди с тем, чтобы выяснить, не пуста ли она. И, если не пуста, то кто «томится» в ней первым – мальчик или девочка? Эта функция не должна изменять состояние очереди.

```
{ P_45_1 - Обработка входного потока,
           Запись в танцевальный кружок }

{ Постановка символа arg в очередь Que }

procedure PutInQue(var Que: string; arg: char);
begin
  Que:= Que + arg;   { добавляем в конец строки }
end;

{ Выбор из очереди Que }

function GetFromQue(var Que: string): char;
begin
  if Length(Que) = 0           { если очередь пуста }
    then GetFromQue:= Char(32) { сообщаем об этом пробелом }
    else begin
           GetFromQue:= Que[1]; { извлекаем первый элемент }
           Delete (Que, 1, 1);  { и удаляем из очереди }
         end
end;

{ Проверка очереди, функция возвращает: Empty, Boy, Girl }

type TQueStatus = (Empty, Boy, Girl); { перечисление }

function TestQue(const Que: string): TQueStatus;
begin
  TestQue:= Empty;             { на случай, если пуста }
  if Length(Que)>0 then
    if Que[1] in ['A'..'Z']
      then TestQue:= Boy       { мальчик }
      else TestQue:= Girl      { девочка }
end;

var S_IN  : string;    { входной поток - символы верхнего и нижнего
регистров }
    S_OUT : string;    { выходной поток }
    Que   : string;    { очередь }
    c_in  : char;      { очередной символ из входного потока }
    i     : integer;   { индекс во входном потоке }

begin
  { задаем входной поток: A..Z - мальчики, a..z - девочки }
  S_IN:='ZHJKqwertASDyuiopQWERTYUIOPasdf';
```

```
  { выходной поток пока пуст }
  S_OUT:='';
  { Очищаем очередь }
  Que:='';

  { Цикл обработки входного потока }
  for i:=1 to Length(S_IN) do begin
    c_in:= S_IN[i];          { выбираем из входного потока }
    if c_in in ['A'..'Z']
      then { если это мальчик...}
        if TestQue(Que) = Girl   { если в очереди есть девочка }
           { добавляем пару в выходной поток }
           then S_OUT:= S_OUT+ c_in + GetFromQue(Que) + ' '
           { а иначе помещаем мальчика в очередь }
           else PutInQue(Que, c_in)
      else { а если это девочка...}
        if TestQue(Que) = Boy    { если в очереди есть мальчик }
           { добавляем пару в выходной поток }
           then S_OUT:= S_OUT + GetFromQue(Que) + c_in + ' '
           { а иначе помещаем девочку в очередь }
           else PutInQue(Que, c_in)
  end;

  Writeln('Входной поток:' );
  Writeln(S_IN);
  Writeln('Выходной поток:' );
  Writeln(S_OUT);

  if Length(Que)>0 then begin
    Writeln('В очереди остались:' );
    Writeln(Que);
  end;
  Readln;
end.
```

Глава 46

Г) Жители райцентра Бюрократовка дневали и ночевали в очереди за справками. Все потому, что там применяли механический текстовый файл – огромную скрипучую книгу, которая листалась лишь в одном направлении – от начала к концу файла. Если первая буква фамилии очередного посетителя следовала по алфавиту далее, чем у предыдущего, то чиновник продолжал листать страницы с текущей позиции, а иначе открывал на первой и листал с начала. Переход от одной буквы алфавита к другой занимал один час. Так, если буквы следовали в порядке «АБВ», то на выдачу справок тратилось три часа, а если в обратном порядке – «ВБА», – то шесть часов (3+2+1). Если же первые буквы фамилий совпадали, то книгу все равно листали заново, поэтому на «ББВ» тратилось шесть часов. Создайте функцию, принимающую «очередь посетителей» – строку из больших латинских букв – и возвращающую время, необходимое для выдачи всех справок.

```
function CalcTime(const arg: string): integer;
var i : integer;
    Res: integer;
    Chr : char;
begin
  Res:=0;
  for i:=1 to Length(arg) do begin
    if i=1 then begin
      { для первого посетителя открываем книгу с начала }
      Chr:= arg[i];
      Res:= Ord(Chr)-Ord('A')+1;
    end else begin
      if arg[i] > Chr
        then Res:= Res+Ord(arg[i])-Ord(Chr)     { листаем дальше }
        else Res:= Res+Ord(arg[i])-Ord('A')+1; { открываем с начала }
      Chr:= arg[i];  { запоминаем предыдущего посетителя }
    end;
  end;
  CalcTime:= Res;
end;


begin
  Writeln(CalcTime('ABD') );
  Writeln(CalcTime('CBA') );
  Writeln(CalcTime('BBB') );
  Writeln('OK'); Readln;
end.
```

Д) Томясь в очереди, свинопас Гришка нашел способ ускорить выдачу справок путем частичного упорядочения очереди (см. задачу **Г**). Создайте функцию, возвращающую такую частично упорядоченную строку (воспользуйтесь множеством символов). Напишите программу для сравнения времен по условиям задач **Г** и **Д**.

```
function CalcTime(const arg: string): integer;
var i : integer;
    Res: integer;
    Chr : char;
begin
  Res:=0;
  for i:=1 to Length(arg) do begin
    if i=1 then begin
      { для первого посетителя открываем книгу с начала }
      Chr:= arg[1];  { запоминаем предыдущего посетителя }
      Res:= Ord(arg[1])-Ord('A')+1;
    end else begin
      if arg[i] > Chr
        then Res:= Res + Ord(arg[i])-Ord(Chr)     { листаем дальше }
        else Res:= Res + Ord(arg[i])-Ord('A')+1; { открываем вновь }
      Chr:= arg[i];  { запоминаем предыдущего посетителя }
    end;
  end;
  CalcTime:= Res;
end;

function Sort(const arg: string): string;
var S : set of 'A'..'Z';
    R : string;
    i : integer;
    c : char;
begin
  R:='';  S:=[];
  for i:=1 to Length(arg) do begin
    if arg[i] in S then begin
      { когда нельзя добавить очередной элемент }
      for C:='A' to 'Z' do
          if C in S then R:= R+C;
      S:=[ arg[i] ];
    end else begin
      { накопление частичной очереди в множестве }
      S:= S+[ arg[i] ];
    end;
  end;
  { добавляем остаток очереди }
  for C:='A' to 'Z' do if C in S then R:= R+C;
  Sort:= R;
end;

var S1, S2 : string;

begin
  S1:= 'CBADCB';     { исходная несортированная очередь }
  S2:= Sort(S1);     { ABCDBC }
  Writeln(S1);
  Writeln(S2);
  Writeln(CalcTime(S1) );     { = 14 }
  Writeln(CalcTime(S2) );     { = 7 }
  Writeln('OK'); Readln;
end.
```

Глава 47

Б) Программист Ник наловчился запоминать сумму цифр в номерах всех автомобилей, попадавших ему на глаза. Однажды он стал свидетелем происшествия, виновник которого скрылся. Ник сообщил полицейским только сумму цифр в номере нарушителя (сам номер Ник не помнил). Помогите полиции, и напишите программу, распечатывающую все трехзначные номера (от 1 до 999), сумма цифр которых равна **N** (значение **N** вводит пользователь).

```
function GetSumDigits(arg: integer): integer;
var sum: integer;
begin
  sum:=0;
  while arg>0 do begin
    sum:= sum + arg mod 10;
    arg:= arg div 10;
  end;
  GetSumDigits:= sum;
end;

var i, N: integer;

begin
  repeat
    Write('Сумма цифр= '); Readln(N);
    Writeln('Подозреваемые автомобили:');
    for i:=1 to 999 do
      if N=GetSumDigits(i) then Write(i:4);
    Writeln;
  until N=0;
end.
```

Д) В функцию передаются три параметра: 1) число, 2) основание системы счисления, 3) символ цифры. Функция должна возвратить количество вхождений этой цифры в представление числа для указанной системы счисления.

```
function CountDigits(aNumber, aBase: integer; aDigit: char): integer;
var sum, digit: integer;
begin
  case Upcase(aDigit) of
    '0'..'9':  digit:= Ord(aDigit)-Ord('0');
    'A'..'F':  digit:= Ord(aDigit)-Ord('A')+10;
    else digit:=-1;
  end;
  sum:=0;
  while aNumber>0 do begin
    if (aNumber mod aBase) = digit  then Inc(sum);
    aNumber:= aNumber div aBase;
  end;
  CountDigits:= sum;
end;

var Number, Base: integer;
    Digit: char;
```

```
begin
  repeat
    Write('Число= '); Readln(Number);
    if Number=0 then break;
    Write('Основание(2..16)= '); Readln(Base);
    Write('Символ(0..9,A..F)= '); Readln(Digit);
    Writeln('Результат= ', CountDigits(Number, Base, Digit));
  until false;
end.
```

Е) Напечатать все трехзначные числа, цифры которых (в десятичном представлении) различны, например: 123, 702.

```
function Test(arg: integer): boolean;
var digs : set of 0..9;
    digit: byte;
begin
  Test:=true;
  digs:=[];
  repeat
    digit:= arg mod 10;
    arg:= arg div 10;
    if digit in digs then begin Test:=false; Break end;
    digs:= digs+[digit];
  until arg=0;
end;

var N: integer;

begin
  for N:=100 to 999 do
    if Test(N) then Write(N:4);
  Writeln;
  Readln;
end.
```

Ж) Найти все шестизначные счастливые билеты. Счастливыми называют билеты, у которых сумма первых 3-х цифр равна сумме следующих 3-х. Например: 123 411. Напишите булеву функцию, определяющую «счастливость» билета.

```
function IsHappy(arg: longint): boolean;
var H, L, sH, sL, i : integer;
begin
  H:= arg div 1000;
  L:= arg mod 1000;
  sL:= 0;
  for i:=1 to 3 do begin sL:= sL + L mod 10; L:= L div 10 end;
  sH:= 0;
  for i:=1 to 3 do begin sH:= sH + H mod 10; H:= H div 10 end;
  IsHappy:= sL=sH
end;
```

```
var N: longint;

begin
  repeat
    Write('N= '); Readln(N);
    if N<0 then break;
    if IsHappy(N) then Writeln('Счастливый!');
  until false;
end.
```

З) В заморской стране обращались денежные купюры достоинством в 1, 2, 5, 10 и 25 пиастров. Напишите программу для кассового аппарата, определяющую наименьший набор купюр, необходимый для выдачи сдачи на указанную сумму. Например, для сдачи 33 пиастров программа напечатает: 25 + 5 + 2 + 1.

```
procedure PrintTheRest(arg: integer);
const Bondes = [1,2,5,10,25]; { множество купюр }
var   Bond: byte; { текущая купюра }
begin
  Bond:=25;
  Write(arg,' = ');
  while arg>0 do begin
    if arg>=Bond then begin
      Write(Bond);
      arg:=arg-Bond
    end else begin
      { подбор следующей купюры }
      repeat Dec(Bond) until Bond in Bondes;
    end;
    if (arg>0) and (arg>=Bond) then Write('+')
  end;
  Writeln;
end;

var N: integer;

begin
  repeat
    Write('N= '); Readln(N);
    PrintTheRest(N)
  until N=0
end.
```

И) Программа шифрования текстового файла заменяет каждый символ двумя шестнадцатеричными цифрами его кода. Например, три символа ‘**405**’ заменяются на шесть символов ‘**343035**’. Символы разбивки строк не затрагиваются. Напишите программу для зашифровки и расшифровки файла по этой системе.

К) Чтобы усилить шифр предыдущей задачи, выполните вращение преобразованной строки на несколько позиций: влево – при зашифровке, и вправо – при расшифровке (смотрите задачи к главе 44).

```
{ Здесь представлено не полное решение, а основные функции }

{ Вращение строки влево (aShift>0) и вправо (aShift<0) }

procedure RotateString(var aStr: string; aShift: integer);
begin
  if aShift>0 then begin
    while aShift<>0 do begin { вращение влево }
      Dec(aShift);
      aStr:= Copy(aStr,2,Length(aStr)) + Copy(aStr,1,1)
    end
  end else begin
    while aShift<>0 do begin { вращение вправо }
      Inc(aShift);
      aStr:= Copy(aStr,Length(aStr),1) + Copy(aStr,1,Length(aStr)-1)
    end
  end;
end;

{ Преобразование символа в две 16-ричные цифры }

function CharToHex(arg: char): string;
const Digs: string ='0123456789ABCDEF';
begin
  CharToHex:= Digs[1+Ord(arg) div 16] + Digs[1+Ord(arg) mod 16]
end;

{ Преобразование двух 16-ричных цифр в символ }

function HexToChar(const arg: string): char;
var i, n : integer;
begin
  n:=0;
  for i:=1 to 2 do begin
    case arg[i] of
      '0'..'9': n:= 16*n + Ord(arg[i])-Ord('0');
      'A'..'F': n:= 16*n + Ord(arg[i])-Ord('A')+10;
    end
  end;
  HexToChar:= Char(n);
end;

{ Шифрование строки }

function CryptStr(const arg: string): string;
var i: integer;
    S: string;
begin
  S:='';
  for i:=1 to Length(arg) do S:=S+ CharToHex(arg[i]);
  CryptStr:= S;
end;


{ Расшифровка строки }
```

```
function DeCryptStr(arg: string): string;
var S: string;
begin
  S:='';
  while Length(arg)>0 do begin
    S:= S + HexToChar(Copy(arg,1,2));
    Delete(arg,1,2);
  end;
  DeCryptStr:= S;
end;

var S: string;

begin
  S:='12345';          { исходная строка }
  S:= CryptStr(S);     { зашифровали }
  Writeln(S);
  RotateString(S,3);   { сдвинули по кругу на 3 влево }
  Writeln(S);
  RotateString(S,-3); { сдвинули по кругу на 3 вправо }
  Writeln(S);
  S:= DeCryptStr(S);   { расшифровали }
  Writeln(S);
  Readln;
end.
```

Л) Напечатайте все числа, не превышающие 1000, такие, что делятся без остатка на каждую из своих цифр. Например: 24, 36, 184, 612. Определите количество таких чисел.

```
function Test(arg: Longint): boolean;
var digit: integer;
    temp: Longint;
begin
  Test:= true;
  temp:= arg;
  repeat
    digit:= temp mod 10;
    temp:= temp div 10;
    if (digit=0) or ((arg mod digit)<>0) then begin
      Test:= false;
      Break;
    end
  until temp=0;
end;

var K, N : Longint;

begin
  N:=0;
  for K:=1 to 1000 do
    if Test(K) then begin
      Inc(N);
      Writeln(N:3, K:5);
    end;
  Readln;
end.
```

Глава 49

Г) Садовая ограда. Вернувшись с курорта, фермер Лефт обнаружил на своем поле чудесно выросший сад. Для сохранения деревьев он обнес его прямоугольной оградой. Пусть ширина и высота поля заданы константами CX и CY, пустые места обозначены точками, а деревья – звездочками. Засадите поле случайным образом и распечатайте его. Затем найдите левый верхний и правый нижний углы ограды и постройте её символом решетки (ограда должна охватывать деревья, но не выходить за пределы поля). Распечатайте сад с оградой.

```
const CX=60;   CY=20;  { ширина и высота поля }

type  TField = array [1..CX, 1..CY] of Char;

var Field : TField;

{ Посадка деревьев }
procedure MakeGarden(N: integer);
var i, x, y: integer;
begin
  FillChar(Field, SizeOf(Field),'.');
  for i:=1 to N do begin
    x:= 1 + Random(CX);
    y:= 1 + Random(CY);
    Field[x,y]:= '*';
  end;
end;

{ Показ поля }
procedure ExpoGarden;
var x, y: integer;
begin
  for y:=1 to CY do begin
    for x:=1 to CX do Write(Field[x,y]);
    Writeln;
  end;
end;

{ Исследование сада }
procedure ScanGarden(var Xmin, Ymin, Xmax, Ymax : integer);
var x, y: integer;
begin
  Xmin:= CX+1;  Ymin:= CY+1;
  Xmax:= 0;     Ymax:= 0;
  for x:=1 to CX do
    for y:=1 to CY do
      if Field[x,y]='*' then begin
         if XMin>x then XMin:=x;
         if YMin>y then YMin:=y;
         if XMax<x then XMax:=x;
         if YMax<y then YMax:=y;
      end;
end;
```

```
{ Постройка ограды }
procedure MakeRail(Xmin, Ymin, Xmax, Ymax : integer);
const CRail='#';
var x, y: integer;
begin
  if YMin>1  then for x:=XMin to XMax do Field[x,Ymin-1]:= CRail;
  if YMax<CY then for x:=XMin to XMax do Field[x,Ymax+1]:= CRail;
  if XMin>1  then for y:=YMin to YMax do Field[XMin-1,y]:= CRail;
  if XMax<CX then for y:=YMin to YMax do Field[XMax+1,y]:= CRail;
end;

var Xmin, Ymin, Xmax, Ymax : integer;  { координаты углов ограды }

begin
  Randomize;
  MakeGarden(10);
  ScanGarden(Xmin, Ymin, Xmax, Ymax);
  Writeln('Xmin, Ymin = ', Xmin:4, Ymin:4);
  Writeln('Xmax, Ymax = ', Xmax:4, Ymax:4);
  ExpoGarden;
  Readln;
  MakeRail(Xmin, Ymin, Xmax, Ymax);
  ExpoGarden;
  Readln;
end.
```

Глава 50

Д) Домино. В этой игре используют 28 костяшек, каждая из которых содержит пару чисел от 0 до 6. Например: 0:0, 1:5, 6:6. Представьте костяшку записью, а игральный набор – массивом этих записей. Заполните массив костяшек и распечатайте его. «Смешайте» костяшки случайным образом и вновь распечатайте массив. Для удобства направьте распечатку в текстовый файл.

```
const CSize= 7*4;
type  TDomino = record N1, N2 : integer end;
      TPack = array [1..CSize] of TDomino;
var   Pack : TPack;

procedure Generate; { Заполнение массива костяшек }
var i, j, k: integer;
begin
  k:=0;
  for i:=0 to 6 do
    for j:=i to 6 do begin
      Inc(k);
      Pack[k].N1:=i;
      Pack[k].N2:=j;
    end;
end;

procedure Expo; { распечатка костяшек }
var k: integer;
begin
  for k:=1 to CSize do Writeln(Pack[k].N1:2,' :', Pack[k].N2:2);
  Readln;
end;

procedure Shuffle; { перемешивание массива костяшек }
var i, j, k: integer;
    temp: TDomino;
begin
  for k:=1 to 100 do begin
    i:= Random(CSize)+1;
    repeat j:= Random(CSize)+1 until j<>i;
    temp:= Pack[i];
    Pack[i]:= Pack[j];
    Pack[j]:= temp;
  end;
end;

begin
  Generate; { Заполнение массива костяшек }
  Expo;
  Shuffle; { Перемешивание массива костяшек }
  Expo;
end.
```

E) Карты. Колода содержит 36 карт четырех мастей: трефы и пики – черные, бубны и червы – красные. Относительная сила карты определяется числом от 6 до 14. Представьте карту записью, содержащей её масть, цвет и силу. Представьте колоду массивом записей, сформируйте полную колоду и распечатайте в текстовый файл. «Перетасуйте колоду и вновь распечатайте в файл. При распечатке учтите, что сила карт от 11 до 14, печатается названиями: валет, дама, король, туз.

```
const CSize= 4*9; { 36 карт в колоде }
type  TCard = record
        mLevel : integer; { 6,7,8,9,10,Валет, Дама, Король, Туз }
        mColor : boolean; { Черная (false) или Красная (true) }
        mSuit  : String;  { Трефы, Бубны, Пики, Червы }
      end;
      TPack = array [1..CSize] of TCard;
var   Pack : TPack;

procedure Generate;
var i, j, k: integer;
begin
  k:=0;
  for i:=0 to 3 do
    for j:=6 to 14 do begin
      Inc(k);
      Pack[k].mLevel:= j;
      if (i mod 2)<>0
           then Pack[k].mColor:=true
           else Pack[k].mColor:=false;
      case i of
        0: Pack[k].mSuit:=' Трефы (Clubs)';
        1: Pack[k].mSuit:=' Бубны (Diamonds)';
        2: Pack[k].mSuit:=' Пики  (Spades)';
        3: Pack[k].mSuit:=' Червы (Hearts)';
      end
    end;
end;

procedure Expo;
var k: integer;
begin
  for k:=1 to CSize do begin
    case Pack[k].mLevel of
      6..10: Write(Pack[k].mLevel:10);
         11: Write('Валет':10);
         12: Write('Дама':10);
         13: Write('Король':10);
         14: Write('Туз':10);
    end;
    if Pack[k].mColor
     then Write('Красная':10)
     else Write('Черная':10);
    Writeln(Pack[k].mSuit);
  end;
  Readln;
end;
```

```
procedure Shuffle;
var i, j, k: integer;
    temp: TCard;
begin
  for k:=1 to 100 do begin
    i:= Random(CSize)+1;
    repeat j:= Random(CSize)+1 until j<>i;
    temp:= Pack[i];
    Pack[i]:= Pack[j];
    Pack[j]:= temp;
  end;
end;

begin
  Generate;
  Expo;
  Shuffle;
  Expo;
end.
```

Глава 52

B) Ник обожал музыку. Но компьютерный музыкальный проигрыватель раздражал программиста, поскольку при случайном выборе мелодий повторял одни песни, напрочь «забывая» о других. Предположим, в списке 10 песен, но звучали только три из них: 3, 6, 5, 6, 3, 6, 5 и т.д.

Ник создал «справедливый» проигрыватель, который выбирал мелодии иначе. Все песни состояли в одном из двух списков: «белом» или «черном». Изначально все они были в «белом» списке, и очередная мелодия выбиралась из него случайно, а после проигрывания ставилась в конец «черного». Если в этот момент в «черном» списке состояла половина мелодий, то первая мелодия из «черного» списка возвращалась в «белый». Затем снова случайно выбиралась мелодия из «белого» списка. Так гарантировалось отсутствие повторов ранее проигранных песен в течение достаточно длительного времени. Создайте программу, генерирующую случайные числа (мелодии) в диапазоне от 1 до **N** представленным выше методом. Значение **N** не превышает 255.

```
var N : integer;              { количество мелодий не более 255 }
    ListB, ListW : string;    { черный и белый списки }
    i : integer;              { индекс }
    R : integer;              { текущий результат выбора }
    Q : string;               { для выхода из цикла }

begin
  ListW:='';  ListB:='';
  Write('N= '); Readln(N);
  { Изначально все песни в белом списке}
  for i:=1 to N do ListW:=ListW+ Chr(i);
  repeat
    i:= Random(Length(ListW))+1;  { случайный индекс в белом списке }
    R:= Ord( ListW[i] );          { выбираем из белого и печатаем }
    Write(R);
    Delete(ListW, i, 1);          { удаляем из белого }
    ListB:= ListB + Char(R);      { и добавляем в конец черного }
    { если в черном уже половина мелодий,
      то первую из черного переносим в белый }
    if Length(ListB) >= N div 2 then begin
      ListW:= ListW + ListB[1];
      Delete(ListB, 1, 1);
    end;
    Readln(Q);          { для выхода из цикла вводим любой символ }
  until Length(Q)>0;
end.
```

Г) Распечатывая числовое множество, мы выводили все его элементы по одному, не заботясь об экономии бумаги или места на экране. Напишите «экономную» процедуру печати множества, которая должна учитывать подряд идущие диапазоны чисел. Вот примеры желаемой распечатки:

```
1,5..255
0..200,210..255
0..255
2,5,7,10..20,30..40
```

```
type TNumbers = set of 0..255;

procedure PrintSet(arg: TNumbers);
var N: integer;
    start, cnt: integer;
    Flag, IsFirst: boolean;
begin
  N:=0; cnt:=0; IsFirst:=true;
  for N:=0 to 255 do begin
    Flag:= N in arg;         { признак обнаружения числа }
    if Flag then begin
      if cnt=0 then start:=N;
      Inc(cnt);
    end;
    { Если очередного числа нет, или все проверены
      и было обнаружено хотя бы одно число... }
    if (not Flag or (N=255)) and (cnt>0) then begin
        { ... то печатаем }
        if not IsFirst then Write(',');  { запятую }
        IsFirst:=false;
        Write(start);                    {  1-е число }
        if cnt>1
          then Write('..',start+cnt-1);  { 2-е число }
        cnt:=0;
    end;
  end;
  Writeln;
end;

begin
  PrintSet([1,5..255]);
  PrintSet([0..200, 210..255]);
  PrintSet([0..255]);
  PrintSet([]);
  PrintSet([5,10..20,30]);
  PrintSet([7]);
  PrintSet([2,5,7,10..20,30..40]);
  Readln;
end.
```

Глава 53

Г) «Глупый» винчестер (об «умном» узнаете в задаче 54-Д). Рассмотрим очень упрощенную модель винчестера, быстродействие которого в значительной степени определяется частотой вращения диска и скоростью перемещения головки чтения-записи. Время одного оборота диска примем за единицу - квант. За это время головка полностью читает или записывает одну дорожку. Количество дорожек на диске – 256, а нумерация идет с нуля (0...255). Время для перемещения головки на соседнюю дорожку тоже примем равным одному кванту.

Винчестером управляет контроллер, работающий несравнимо быстрее механических узлов - диска и головки. Поэтому издержками времени на его работу пренебрежем. Через известный интервал времени контроллер просматривает входную очередь, содержащую запросы на чтение или запись дорожек. Эта очередь формируется всеми запущенными программами. Мы заменим её текстовым файлом, каждая строка которого содержит по несколько чисел в диапазоне от 0 до 255 – это номера запрашиваемых дорожек. Пустая строка говорит об отсутствии запросов в текущий момент времени. Для первой строки файла сделаем исключение, поместив там лишь одно число - период просмотра этой очереди.

Контроллер «рулит» так. Прочитав список запросов (очередную строку файла), он перемещает их в свою внутреннюю очередь и далее обрабатывает её в порядке получения запросов: смещает головку в нужную позицию и выполняет чтение-запись. По ходу этой работы он следит за таймаутом, и, по истечении оного, читает следующую порцию входной очереди (строку файла). Ваша программа должна подсчитать общее время обработки запросов для набора данных из входного файла.

```
var F: Text;  { входной файл,
                в первой строке - период опроса входной очереди,
                в последующих - списки запросов }
    Period: integer;         { период опроса входной очереди }
    TimeOut: integer;        { таймаут чтения входной очереди }
    Track: integer;          { текущий запрос из внутренней очереди }
    Position: integer;       { текущая позиция головки }
    ProgramResult: integer; { общее время работы программы }
{-------------------------------------------------------}
{ Постановка в очередь и извлечение из нее }

var Que: string;  { очередь }

procedure PutInQue(aTrack: integer); { Постановка в очередь }
begin
  Que:= Que + Char(aTrack);   { добавляем в конец строки}
end;

function GetFromQue: integer;  { Выбор из очереди }
begin
  if Length(Que) = 0             { если очередь пуста }
    then GetFromQue:= -1
    else begin
      GetFromQue:= Ord(Que[1]); { возвращаем первый элемент }
      Delete (Que, 1, 1);       { и удаляем его из очереди }
    end
```

```
end;
{-------------------------------------------------------------}
{ Проверка истечения таймаута
 и чтение очередной порции запросов из строки файла }
procedure TimeOutHandler;
var N: integer;
begin
  if TimeOut>0 then begin
     Inc(ProgramResult);    { общее время }
     Dec(TimeOut);
  end
  else begin
      TimeOut:= Period;
     { Если истек таймаут, читаем порцию из файла }
      if not EoF(F) then begin
        while not Eoln(F) do begin
          Read(F, N);
          PutInQue(N)
        end;
        Readln(F);
      end;
  end;
end;
{-------------------------------------------------------------}
{ Обработка запроса на чтение-запись дорожки }
procedure QueryHandler (aTrack: integer);
begin
  { Write(aTrack:4); }
  { продвигаем головку в требуемую позицию }
  while Position<>aTrack do begin
    if Position<aTrack
      then Inc(Position)
      else Dec(Position);
    TimeOutHandler;          { таймаут чтения из файла }
  end;
  { Один квант тратим на чтение-запись }
  TimeOutHandler;         { таймаут чтения из файла }
end;
{-------------------------------------------------------------}
begin   { Main }
  ProgramResult:=0;      { Общее время }
  Position:=0;           { позиция головки }
  TimeOut:= 0;           { таймаут }
  Que:='';               { внутренняя очередь пуста }
  Assign(F, 'Disk.in'); Reset(F);
  Readln(F,Period);      { в первой строке - период опроса очереди }
  repeat
    Track:= GetFromQue;   { извлекаем из внутр. очереди }
    if Track>0
      then QueryHandler(Track)  { выполнить запрос }
      else if Eof(F)     { если входной файл пуст }
             then Break { то выход }
             else TimeOutHandler;{ а иначе отрабатываем таймаут
                                   и читаем строку файла }
  until false;
  Write('Result= ',ProgramResult); Readln;
end.
```

Глава 54

Д) В задаче 53-Г была представлена модель «глупого» винчестера. «Умный» винчестер отличается организацией внутренней очереди и челночным движением головки. Она следует попеременно то от внутренней дорожки к внешней, то обратно, попутно выполняя все накопившиеся в очереди запросы. Направление движения переключается, когда в текущем направлении не остается запросов, поэтому головка редко достигает крайних дорожек.

Ваша программа должна подсчитать общее время обработки запросов «умным» контроллером для набора данных из входного файла, составленного по правилам для задачи 53-Г. Создайте несколько наборов таких данных и сравните время их обработки двумя типами контроллеров.

Подсказка. Для организации внутренней очереди контроллера здесь можно применить массив чисел (счетчиков). Каждый счетчик будет хранить количество запросов для своей дорожки. При постановке запроса в очередь счетчик наращивается, а при извлечении уменьшается.

```
var F: Text;  { входной файл,
                в первой строке - период опроса входной очереди,
                в последующих - списки запросов }
    Period: integer;         { период опроса входной очереди }
    TimeOut: integer;        { таймаут чтения входной очереди }
    Track: integer;          { текущий запрос из внутренней очереди }
    Position: integer;       { текущая позиция головки }
    Direction: integer;      { направление движения = +1 / -1 }
    ProgramResult: integer; { общее время работы программы }
{-------------------------------------------------------}
{ Постановка в очередь и извлечение из нее }

const CTracks = 256;  { количество дорожек }
type  TTracks = array[0..CTracks-1] of integer;
var   Tracks: TTracks; { массив счетчиков для дорожек }

procedure PutInQue(aItem: integer);
begin
  { наращиваем счетчик дорожки }
  Inc(Tracks[aItem]);
end;

{ Выбор из очереди }

function GetFromQue(aPos, aDirection: integer): integer;
var i: integer;
begin
  i:= aPos;
  while (i in [0..CTracks-1]) and (Tracks[i]=0)
    do i:=i+aDirection;  { +1/-1 }
  if i in [0..CTracks-1]
    then begin
           GetFromQue:= i;
           Dec(Tracks[i])
         end
```

```
    else GetFromQue:=-1;
end;

{-----------------------------------------------------------}
{ Проверка истечения таймаута
 и чтение очередной порции запросов из строки файла }

procedure TimeOutHandler;
var N: integer;
begin
 if TimeOut>0 then begin
     Dec(TimeOut);
     Inc(ProgramResult);
   end
   else begin
      TimeOut:= Period;
     { Если истек таймаут, читаем входную очередь (файл)}
      if not Eof(F) then begin
        while not Eoln(F) do begin
          Read(F, N);
          PutInQue(N)
        end;
        Readln(F);
      end;
  end;
end;
{-----------------------------------------------------------}
{ Обработка запроса на чтение-запись дорожки }

procedure QueryHandler(aTrack: integer);
begin
  { Write(aTrack:5); }
  { Продвижение к дорожке }
  while Position<>aTrack do begin
    if Position<aTrack
      then Inc(Position)
      else Dec(Position);
    TimeOutHandler;
  end;
  { Чтение-запись дорожки + 1 квант }
  TimeOutHandler;
end;
{-----------------------------------------------------------}
begin   { Main }
  ProgramResult:=0;      { Общее время }
  Position:=0;           { позиция головки }
  TimeOut:= 0;           { таймаут }
  Direction:=+1;         { начальное направление движения головки }
  FillChar(Tracks, SizeOf(Tracks), 0);
  Assign(F, 'Disk.in'); Reset(F);
  Readln(F, Period);     { в первой строке - период опроса очереди }
  repeat
    Track:= GetFromQue(Position, Direction);   { извлекаем запрос }
    if Track>=0
      then QueryHandler(Track) { обрабатываем }
      else if Eof(F)           { если входной файл прочитан }
             then Break        { то выход }
             else begin
```

```
                TimeOutHandler;{ а иначе таймаут и читаем строку файла
}
                Direction:= -Direction; { меняем направление движения }
              end;
  until false;
  Write('Result= ',ProgramResult); Readln;
end.
```

Глава 56

Б) В главе 45 было высказано предположение, что для записи в танцевальный кружок достаточно одной очереди. Докажите это, создав соответствующую программу. Чем нужно дополнить процедуры работы с очередью?

```
type  PRec = ^TRec;          { Тип указатель на запись }
      TRec = record          { Тип запись для хранения связанных строк
}
        mStr  : string[31]; { хранимая строка (имя) }
        mBoy  : boolean;    { true - мальчик, false - девочка }
        mNext : PRec;       { указатель на следующую запись }
      end;

  { Процедура размещения строки в очереди }

procedure PutInQue(var Que: PRec; const arg: string; aBoy: boolean);
var p: PRec;
begin
  New(p);              { создаем новую переменную-запись }
  p^.mStr:= arg;       { размещаем строку }
  p^.mBoy:= aBoy;      { признак = мальчик (true) или девочка (false) }
  { размещаем указатель в голове очереди }
  p^.mNext:= Que;      { указатель на предыдущую запись }
  Que:=p;              { текущая запись в голове очереди }
end;

     { Извлечение строки из начала очереди (из конца списка) }

function GetFromQue(var Que: PRec): string;
var p1, p2: PRec;
begin
  GetFromQue:= '';
  if Assigned(Que) then begin
    { Поиск первого элемента очереди }
    p1:= Que;  p2:=p1;
    { если в очереди один элемент, цикл не выполнится ни разу! }
    while Assigned(p1^.mNext) do begin
      p2:=p1;           { текущий }
      p1:=p1^.mNext;    { следующий }
    end;
    { теперь p1 указывает на первый элемент очереди,
      а p2 - на второй (или на тот-же самый,
      если в очереди всего один элемент)}
    GetFromQue:= p1^.mStr;   { извлекаем данные }
    if p1=p2                 { если в очереди был один элемент... }
      then Que:= nil  { очередь стала пустой }
      else p2^.mNext:= nil;  { а иначе "отцепляем" первый элемент }
    Dispose(p1);             { освобождаем память первого элемента }
  end;
end;
```

```
{ Проверка очереди, функция возвращает: Empty, Boy, Girl }

type TQueStatus = (Empty, Boy, Girl); { перечисление }

function TestQue(Que: PRec): TQueStatus;
begin
  TestQue:= Empty;
  if Assigned(Que) then
    if Que.mBoy
      then TestQue:= Boy
      else TestQue:= Girl
end;

var
    Que    : PRec;        { очередь }
    S_in   : String;      { строка с именем }
    FlagBoy: boolean;     { признак чтения имени мальчика }
    F_In, F_Out : Text;   { входной и выходной файлы }

begin
  Que  := nil ; { Очищаем очередь }

  Assign(F_In, 'P_56_2.in');  Reset(F_In);
  Assign(F_Out,'P_56_2.out'); Rewrite(F_Out);

  { Цикл обработки входного потока }

  while not Eof(F_In) do begin
    Readln(F_In, S_in);   { выбираем имя из входного потока }
    FlagBoy:= S_in[1]<>' '; { строки девочек начинаются с пробела! }
    while S_in[1]=' ' do Delete(S_in,1,1);
    if FlagBoy
      then { если это мальчик...}
        if TestQue(Que)=Girl   { если в очереди есть девочка }
          { добавляем пару в выходной поток }
          then Writeln(F_Out,S_in+' + '+GetFromQue(Que))
          { а иначе помещаем мальчика в очередь }
          else PutInQue(Que, S_in, true)
      else { а если это девочка...}
            if TestQue(Que)= Boy  { если в очереди есть мальчик }
              { добавляем пару в выходной поток }
              then Writeln(F_Out,GetFromQue(Que)+' + '+S_in)
              { а иначе помещаем девочку в очередь }
              else PutInQue(Que, S_in, false)
  end;
  Close(F_In);  Close(F_Out);
end.
```

Глава 59

Б) Императорские заботы. После постройки империи (главы 57 и 58), бывшие независимые государства стали её провинциями и породили новые проблемы. Для доставки туда важных правительственных распоряжений император нанял гонцов. Чтобы доставка была по возможности скорой, гонцы следовали лишь в одном направлении – от центра к окраинам империи. Сколько гонцов для этого нужно? – вот первый вопрос. Сколько времени потребуется для достижения самых дальних окраин, если переход из провинции в провинцию отнимает сутки? – это второй вопрос. На окраинах нужны гостиницы для отдыха гонцов, что это за окраины? Определите их названия.

Подсказка: возьмите за основу программу «P_58_1» – обход графа в ширину – и сделайте необходимые изменения в процедуре **Expand**.

```
{ Измененная процедура расширения (экспансии) "империи",
  вдобавок вычисляются:
- количество гонцов, требуемых для охвата империи
- расстояние до самой удаленной окраины
- перечень окраин империи
}
type TNames = set of char;  { тип для названий - множество символов }

procedure Expand(arg : PNode;
                 var aRunners: integer; { кол-во гонцов }
                 var aMaxDist: integer; { максим. расстояние }
                 var aPeripher: TNames  { окраины империи }
                );
var p : PNode;
    q : PLink;
    flag: boolean; { вспомогательный флаг }
begin
  aRunners:=0;     { кол-во гонцов }
  aMaxDist:=0;     { максим. расстояние }
  aPeripher:=[];   { окраины империи }

  arg^.mDist:= 0;            { расстояние до центра империи = 0 }
  arg^.mColor:= Gray;        { метим серым цветом }
  PutInQue(arg);             { и помещаем в очередь обработки }
  while GetFromQue(p) do begin     { извлекаем очередной узел }
    Write(p^.mName, ' ->');        { печатаем название узла для
наблюдения }
    q:= p^.mLinks;                 { начинаем просматривать соседей }
    flag:=false;
    while Assigned(q) do begin
      if q^.mLink^.mColor = White then begin  { если сосед еще белый }
        q^.mLink^.mColor:= Gray;          { метим его серым }
        q^.mLink^.mDist:= p^.mDist +1;    { расстояние к центру }
        q^.mLink^.mPrev:= p;              { откуда пришли }
        PutInQue(q^.mLink);               { и помещаем в очередь обраб.}
        Write(q^.mLink^.mName:2);         { печатаем для аблюдения }
        flag:= true;                      { это не окраина империи }
        if aMaxDist < q^.mLink^.mDist
          then aMaxDist:= q^.mLink^.mDist;  { максим. расстояние }
      end;
      q:= q^.mNext;     { переход к следующему соседу }
```

```
    end;
    { Если флаг не установлен, значит белых узлов рядом нет,
     и значит это окраина империи  }
    if not flag then begin
      Inc(aRunners);        { увеличиваем количество гонцов }
      aPeripher:= aPeripher + [ p.mName ]; { добавляем к окраинам }
    end;
    p^.mColor:= Black; { после обработки узла метим его черным }
    Writeln;           { для наблюдения }
  end;
end;

var     F_In {, F_Out} : Text;  { входной и выходной файла }
        C : Char;         { название страны }
        Start : PNode;    { узел, с которого расширяется "империя" }

        Runners: integer; { кол-во гонцов }
        MaxDist: integer; { расстояние до самой удаленной окраины }
        Peripher: TNames; { окраины империи }

begin   { Главная программа }

  { Инициализация списка узлов и очереди узлов }
  List:= nil;  Que:= nil;
  Assign(F_In, 'P_57_1.in');
  ReadData(F_In);              { чтение графа }
  { Цикл ввода названий стран }
  repeat
    Write('Name= '); Readln(C);
    C:= UpCase(C);
    if not (C in ['A'..'Z']) then break;
    Start:= GetPtr(C);             { центр империи }
    if Assigned(Start) then begin { если такая страна существует, }
      InitList;  { устанавливаем начальные значения в полях узлов }
      { расширяем "империю" от центра Start }
      Expand(Start, Runners, MaxDist, Peripher);
      Writeln('Количество гонцов: ', Runners);
      Writeln('Максимальное расстояние: ', MaxDist);
      Write('Окраины империи: ');
      for C:= 'A' to 'Z' do if C in Peripher then Write(C:2);
      Writeln;
    end;
  until false
end.
```

Глава 60

Б) Контрразведка перехватила несколько зашифрованных файлов, подозревая, что это тексты написанных на Паскале вирусов. Позвали Шерлока Ивановича Холмского в надежде, что тот расшифрует их. Шерлок Иванович предположил, что шифровали методом Юлия Цезаря (вспомните главу 24). Нужен ключ! После недолгих раздумий Шерлок Иванович создал программу для подбора ключей к таким текстам. Повторите еще один «подвиг контрразведчика», или слабо? Подсказка: в таких файлах после расшифровки обязательно встречаются ключевые слова **BEGIN** и **END** – воспользуйтесь этим.

```
const CInName='CRYPT.TXT';  { имя зашифрованного файла }
      COutName='CRYPT.OUT'; { имя расшифрованного файла }

{ Дешифрование одного символа }

function DeCryptChar(arg: char; key: integer): char;
var x: integer;
begin
  DeCryptChar:=arg;
  if Ord(arg)>=32 then begin { управляющие символы не трогаем }
    x:= Ord(arg)- key;
    if x<32 then x:= x+256-32;
    DeCryptChar:= Char(x);
  end;
end;

{ Дешифрование строки }

procedure DeCryptString(var arg: string; key: integer);
var k: integer;
begin
  for k:=1 to Length(arg) do arg[k]:= DeCryptChar(arg[k], key);
end;

{----- Процедура дешифрования файла -----}

procedure  DeCryptFile(key: integer);
var  FileIn:  text;   { входной файл для чтения }
     FileOut: text;   { выходной файл для записи }
     S:    string;
begin
     Assign(FileIn, CInName);
     Assign(FileOut, COutName);
     Reset(FileIn);        { открыть входной файл для чтения }
     Rewrite(FileOut);     { открыть выходной файл для записи }
     while not Eof(FileIn) do begin
          Readln(FileIn, S);   { читать очередную строку }
          DeCryptString(S, key); { дешифровать }
          Writeln(FileOut, S); { записать в выходной файл }
     end;
     { закрыть оба файла }
     Close(FileIn);  Close(FileOut);
end;
```

```
procedure UpCaseStr(var arg: string);
var i: integer;
begin
  for i:=1 to Length(arg) do  arg[i]:= UpCase(arg[i]);
end;

{ Сканирование файла в поисках ключа }

function ScanKey: integer;
var   FileIn:  text;  { входной файл для чтения }
      S: string;
      i: Integer;
      key: Integer;
      IsBegin, IsEnd : boolean;
begin
      key:=-1; IsBegin:= false; IsEnd:= false;
      Assign(FileIn, CInName);
      for i:=0 to 255 do begin { перебор всех ключей }
      Reset(FileIn);
      { просмотр файла в поисках BEGIN и END }
      while not Eof(FileIn) and (key<0) do begin
        Readln(FileIn, S);
        DeCryptString(S,i); { i - текущий ключ шифра }
        UpCaseStr(S);       { приводим к верхнему регистру }
        IsBegin:= IsBegin or (Pos('BEGIN',S)<>0);
        IsEnd:= IsEnd or (Pos('END',S)<>0);
        if IsBegin and IsEnd then key:=i; { нашли ключ! }
      end;
      if key>=0 then break; { нашли ключ! }
      end;
      ScanKey:= key;
end;

      {----- Главная программа -----}

var Key: integer; { искомый ключ шифра }

begin
      Key:= ScanKey;
      if Key>=0 then begin
       { нашли ключ! }
       Writeln('Ключ= ', Key);
       DeCryptFile(Key);
      end;
end.
```

B) Рейтинговое голосование. По избирательному закону Иксляндии каждый избиратель голосует не за одного, а за всех кандидатов, включенных в бюллетень, расставляя их в порядке своего предпочтения. Побеждает кандидат, набравший наименьшую сумму мест (если таковых несколько, то проводят второй тур). Предположим, баллотируются четыре кандидата, а бюллетени содержат следующие предпочтения избирателей:

```
3 4 2 1
2 4 3 1
4 1 3 2
```

Здесь первый кандидат набирает сумму 10, второй – 8, третий – 7, четвертый – 5. Таким образом, побеждает четвертый кандидат в списке.

Количество кандидатов известно и равно пяти. Ваша программа принимает файл, каждая строка которого содержит 5 чисел – данные одного бюллетеня. Надо выдать список победителей голосования (одного или нескольких).

```
{ Контрольный пример:
1 2 3 4 5
5 4 3 2 1
1 3 5 2 4
2 1 3 5 4
5 4 3 2 1
---------
14 15 14 18 14 □ суммы мест
}
const CNomin = 5;
      CFileName = 'Nomin.in';
type  TNomArray = array [1..CNomin] of longint;
var   NomArr : TNomArray;

procedure Init; { очистка массива счетчиков }
var i: integer;
begin
  for i:=1 to CNomin do NomArr[i]:=0;
end;

procedure Print; { распечатка счетчиков }
var i: integer;
begin
  for i:=1 to CNomin do Write(NomArr[i]:5);
  Writeln;
end;
```

```
procedure Calc; { подсчет голосов }
var F: Text;
    n: integer; { = 1..5 - место в бюллетене }
    B: integer; { = 1..5 - номер кандидата }
begin
  Assign(F, CFileName);
  Reset(F);
  while not Eof(F) do begin
    n:=0;
    while not Eoln(F) and (n<CNomin) do begin
      Read(F, B);
      Inc(n);
      if B in [1..CNomin] then NomArr[B]:= NomArr[B]+n;
    end;
    Readln(F);
  end;
end;

procedure Handle; { обработка голосов }
var i: integer;
    m: longint;
begin
  m:= $7FFFFFFF; { MaxLongint - максимальное число }
  for i:=1 to CNomin do if m > NomArr[i] then m:=NomArr[i];
  Writeln('Минимальная сумма= ',m);
  Write('Победители: ');
  for i:=1 to CNomin do if m = NomArr[i] then Write(i:3);
  Writeln;
end;

begin
  Init;
  Calc;
  Print;
  Handle;
  Readln;
end.
```