Богомолова Ольга Борисовна,
Усенков Дмитрий Юрьевич

ЗАДАЧИ НА АНАЛИЗ И ОБРАБОТКУ ДАННЫХ: «ВЫСШИЙ ПИЛОТАЖ» ЕГЭ ПО ИНФОРМАТИКЕ[1]

Пропуск текстовых данных из потока символов

1) использование «буферной» переменной соответствующего типа, которая далее не используется (при чтении следующего ненужного данного используется та же переменная);

2) для пропуска текстовых данных в цикле считывать из входной строки отдельные символы, включая пробел, завершающий очередное данное. Для пропуска еще одного такого данного повторно строится такой же цикл.

Массивы-счетчики

Для подсчета количеств нескольких объектов используется *массив-счетчик* размера, соответствующего количеству объектов. При обнаружении в потоке входных данных объекта, соответствующего условию, нужно увеличить на 1 значение ячейки массива-счетчика, соответствующей данному объекту.

Индексные массивы

Если в качестве индексов массива-счетчика невозможно (или не рационально) использовать значения, считываемые из входных данных в качестве идентификатора объектов, то применяется *индексный массив*:

1) массив-счетчик объявляется по количеству обрабатываемых объектов;

2) отдельно объявляется массив того же размера, который содержит обозначения соответствующих объектов;

3) при считывании входных данных соответствующее обозначение объекта ищется в индексном массиве (путем его просмотра с начала до конца); номер ячейки индексного массива, содержащей искомое обозначение объекта, запоминается;

4) если данный объект соответствует требуемому условию, то в массиве-счетчике увеличивается на 1 элемент, индекс которого равен запомненному индексу элемента в индексном массиве, содержащем обозначение этого объекта.

[1] Окончание. Начало в № 5/2018.

РАЗБОР ТИПОВЫХ ЗАДАЧ

Задача 1[2]. На вход программе подаются сведения о номерах школ учащихся, участвовавших в олимпиаде. В первой строке сообщается количество учащихся *N*, каждая из следующих *N* строк имеет формат: **<Фамилия> <Инициалы> <номер школы>**, где **<Фамилия>** – строка, состоящая не более чем из 20 символов, **<Инициалы>** – строка, состоящая из четырех символов (буква, точка, буква, точка), **<номер школы>** – не более чем двузначный номер. **<Фамилия>** и **<Инициалы>**, а также **<Инициалы>** и **<номер школы>** разделены одним пробелом.

Пример входной строки:

```
Иванов П.С. 57
```

Требуется написать как можно более эффективную программу (укажите используемую версию языка программирования, например Borland Pascal 7.0), которая будет выводить на экран информацию, из какой школы было меньше всего участников (таких школ может быть несколько). При этом необходимо вывести информацию только по школам, пославшим хотя бы одного участника.

Следует учитывать, что $N \geq 1000$.

Решение

1. По условию номер школы – не более чем двузначный, тогда максимальное значение номера школы равно 99. Требуется произвести подсчет количества участников от каждой школы, следовательно, требуется массив-счетчик с индексами от 1 до 99.

```
var nc: array[1..99] of integer;
```

Необходимость использовать индексный массив отсутствует.

2. Входные данные: вначале считывается количество строк *N*, затем в цикле с параметром считываются сами эти строки.

Фамилия и инициалы участника – лишние данные, их нужно пропустить путем посимвольного чтения до завершающего пробела включительно.

Номер школы считывается в переменную целого типа.

3. Обработка считанных данных: увеличиваем на 1 значение элемента массива-счетчика, индекс которого равен считанному номеру школы.

4. Обработка полученного массива-счетчика: стандартный алгоритм поиска минимума в одномерном массиве для ненулевых элементов.

5. Вывод данных: просмотр всего массива-счетчика и вывод индексов его элементов, значения которых равны найденному минимуму.

Полный текст программы:

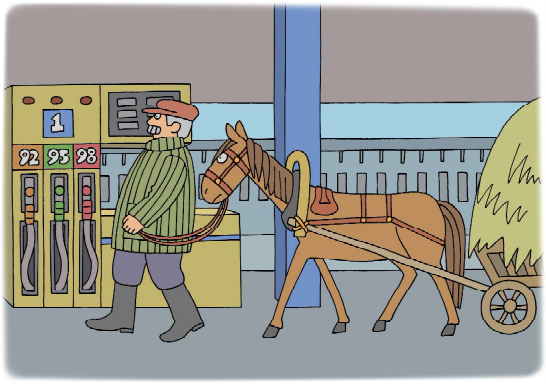
```
var nc: array[1..99] of integer;
    p: 1..99;
    c: char;
    i, k, N, min: integer;
begin
  for i := 0 to 99 do nc[i]:=0;  // обнуление массива-счетчика
  readln(N);  // считано количество строк
```

[2] С4/ДЕМО2009.

```
  for i := 1 to N do  // чтение строк данных
   begin
    repeat
      read(c)
    until c=' '; // пропущена фамилия
    repeat
      read(c)
    until c=' '; // пропущены инициалы
    readln(p);   // считан номер школы
    nc[p]:=nc[p]+1;  // увеличение счетчика для данной школы
  end;

// поиск минимума
  min:=N;
  // максимально возможное значение количества участников – N, если все
     они от одной и той же школы
  for i:=1 to 99 do
    if (nc[i]>0) and (nc[i]<min) then min:=nc[i];
   // если текущее значение больше нуля и меньше предполагаемого минимума,
      то назначить его как новый минимум
  // вывод результатов
  for i:=1 to 99 do
    if nc[i]=min then
      writeln(i);
end.
```

Задача 2[3]. На автозаправочных станциях (АЗС) продается бензин с маркировкой 92, 95 и 98. В городе *N* был проведен мониторинг цены бензина на различных АЗС.



Напишите эффективную по времени работы и по используемой памяти программу (укажите используемую версию языка программирования, например Borland Pascal 7.0), которая будет определять для каждого вида бензина, сколько АЗС продают его дешевле всего. На вход программе в первой строке подается число данных о стоимости бензина. В каждой из последующих *N* строк находится информация в следующем формате:

<Компания> <Улица> <Марка> <Цена>,

где **<Компания>** – строка, состоящая не более чем из 20 символов без пробелов, **<Улица>** – строка, состоящая не более чем из 20 символов без пробелов, **<Марка>** – одно из чисел – 92, 95 или 98, **<Цена>** – целое число в диапазоне от 1000 до 3000, обозначающее стоимость одного литра бензина в копейках. **<Компания>** и **<Улица>**, **<Улица>** и **<Марка>**, а также **<Марка>** и **<Цена>** разделены ровно одним пробелом.

Пример входной строки:

```
Синойл Цветочная 95 2250
```

Программа должна выводить через пробел 3 числа – количество АЗС, продающих дешевле всего 92-й, 95-й и 98-й бензин соответственно. Если бензин какой-то марки нигде не продавался, то следует вывести 0.

[3] С4/ДЕМО2010.

Пример выходных данных:

```
12 1 0
```

Решение

1. Необходимо определять минимальную цену на бензин каждого вида и количество таких значений. Это можно делать сразу при обработке введенных данных без организации массива-счетчика.

Вместо этого удобно определить два массива (взаимосвязанных через индексы элементов), в элементах которых будут храниться соответственно текущее минимальное значение цены бензина соответствующей марки (массив **min**) и количество элементов, равных этому минимуму (массив **ans**).

```
var min, ans: array[92..98] of integer;
```

При этом принципы работы с этими массивами аналогичны работе с массивом-счетчиком: считанный номер марки бензина используется как индекс соответствующего элемента в массиве. (Массивы объявлены на 7 элементов каждый, при этом в каждом используется только по три элемента.)

2. Входные данные: сначала отдельно считывается количество строк *N*, а затем в цикле с параметром считываются сами строки.

При этом строковые данные **<Компания>** и **<Улица>** – лишние, их нужно пропустить путем посимвольного чтения до завершающего пробела включительно.

Номер бензина и цена (целые числа) считываются в соответствующие переменные.

3. Обработка считанных данных: считанное значение номера марки бензина используется как индекс и определяет, с какой парой переменных (предполагаемый минимум и счетчик значений, равных ему) мы работаем в данном случае. Считанное значение цены обрабатывается в типовом алгоритме поиска минимума с подсчетом количества значений, равных минимуму, аналогично элементу массива:

1) цена, по условию, не превышает 3000, поэтому при начальной инициализации ячеек массива **min** в качестве предполагаемых минимумов задается константа, заведомо большая максимально возможного значения цены (число 3001 или большее); счетчики количества значений, равных минимальному (массив **ans**), обнуляются.

ВАЖНО! Все эти инициализационные действия производятся в цикле (индексы массивов меняются от 92 до 98) *до* начала цикла чтения входных строк.

2) после чтения номера марки бензина **k** и цены **b** выполняется проверка: если **b < min[k]**, то:

– значение **b** запоминается как новое значение предполагаемого минимума **min[k]**;

– значение счетчика **ans[k]** устанавливается в 1 (то есть одно такое значение уже найдено);

ВАЖНО! В зависимости от номера марки бензина работаем с соответствующей парой ячеек «минимум – счетчик».

3) иначе если значение **b** равно текущему значению минимума для данной марки бензина **min[k]**, то увеличиваем на 1 значение счетчика **ans[k]**.

ВАЖНО! Если бензина какой-то марки не было вообще, то значение соответствующего счетчика не изменится (останется равным нулю).

По завершении обработки входных данных имеем в ячейках **ans[92]**, **ans[95]** и **ans[98]** искомые количества встреченных значений цен, равных минимальным (для каждой из трех этих марок), либо нули.

4. Вывод данных: на экран выводятся полученные значения **ans[92]**, **ans[95]** и **ans[98]**.

Полный текст программы:

```
program C4;
var min, ans: array[92..98] of integer;
    c: char;
    i, k, N, b: integer;
begin
  for i:=92 to 98 do  // инициализация начальных значений
   begin              // предполагаемых минимумов и счетчиков
    min[i]:=3001;
    ans[i]:=0;
   end;

  readln(N);          // считано количество строк
  for i:=1 to N do    // чтение строк входных данных
   begin
    repeat
      read(c);
    until c=' ';      // пропуск названия компании
    repeat
      read(c);
    until c=' ';      // пропуск названия улицы
    readln(k,b);      // чтение номера марки бензина и его цены

   // поиск минимума с подсчетом количества элементов, равных этому минимуму
    if b < min[k] then
     begin
      min[k] := b;
      ans[k] := 1
     end else
    if min[k] = b then ans[k] := ans[k]+1;
  end;

  // вывод результатов
  writeln(ans[92],' ', ans[95],' ', ans[98])
end.
```

Задача 3[4]. На вход программе подается набор символов,заканчивающийся точкой (в программе на языке Бейсик символы можно вводить по одному в строке, пока не будет введена точка, или считывать данные из файла). Напишите эффективную, в том числе и по используемой памяти, программу (укажите используемую версию языка программирования, например Borland Pascal 7.0), которая сначала будет определять, есть ли в этом наборе символы, соответствующие десятичным цифрам. Если такие символы есть, то можно ли их переставить так, чтобы полученное число было симметричным (читалось одинаково как слева направо, так и справа налево). Ведущих нулей в числе быть не должно, исключение – число 0, запись которого содержит ровно один ноль.

[4] С4/ДЕМО2011.

Если требуемое число составить невозможно, то программа должна вывести на экран слово “NO”. А если возможно, то в первой строке следует вывести слово “YES”, а во второй – искомое симметричное число. Если таких чисел несколько, то программа должна выводить максимальное из них. Например, пусть на вход подаются следующие символы:

```
Do not 911 to 09 do.
```

В данном случае программа должна вывести:

```
YES
91019
```

Решение

1. Необходимо вычислять количества каждой из цифр от 0 до 9, поэтому объявляем массив-счетчик с индексами элементов от “0” до “9”.

```
var a: array['0'..'9'] of integer;
```

ВАЖНО! Обратите внимание: индексы элементов массива – это символы “1” .. “9”, а не числа 1 .. 9 (либо при использовании в качестве индексов именно чисел нужно считанные из входной строки символы цифр преобразовывать в их числовые значения).

Поскольку в качестве индексов массива-счетчика используются сами считываемые обозначения объектов, использовать индексный массив не требуется.

2. Входные данные: считываются посимвольно до завершающей точки (цикл с предусловием).

3. Обработка считанных данных: если считанный символ представляет собой цифру, то в массиве-счетчике увеличиваем на 1 значение элемента с индексом, равным этому символу цифры.

4. Обработка полученного массива-счетчика.

В каких случаях можно построить максимальное по величине симметричное число? Очевидно, что это возможно в трех случаях:

– если количества всех цифр четны;

– если есть только одна цифра, количество которой нечетно (цепочка этих цифр будет в полученном симметричном числе стоять в середине);

– если не имеется никаких цифр, кроме нулей, то симметричное число можно построить только если этот нуль – единственный.

Исходя из анализа этих условий и решается задача[5]:

1) просматривая массив-счетчик, подсчитываем в нем (в переменной **k**) количество цифр, встречающихся нечетное число раз. При этом в переменной **c_odd** запоминаем символ цифры, которая последней встретилась нечетное число раз, а в переменной **s** независимо от четности количества каждой цифры подсчитываем суммарное количество всех цифр (оно понадобится позже для проверки другого условия).

```
k := 0;
for c := '0' to '9' do
 begin
  if a[c] mod 2 = 1 then
   begin
    k := k + 1;
    c_odd := c
   end;
  s := s+a[c];
 end;
```

[5] В демонстрационном варианте ЕГЭ за 2011 год приведено другое решение, но оно достаточно сложно для понимания. Предлагаемый здесь вариант решения более «прозрачен» для понимания смысла алгоритма.

2) проверяем условие: количество нулей равно 1, а количества других цифр равны нулю (что понятно из равенства общей суммы количеств всех цифр всё той же единице). Если это истинно, то выводится слово “YES” и число 0.

```
if (a['0'] = 1) and (s = 1) then
 begin
  writeln('YES');
  writeln('0');
 end else
```

3) иначе проверяем условие: если нули – не единственные цифры и при этом **k = 0** или **k = 1**, то выводится слово “YES” и конструируется число; иначе выводится слово “NO”:

```
if (s <> a['0']) and ((k = 0) or (k = 1)) then
 begin
  writeln('YES');
  <вывод симметричного числа>
 end else writeln('NO');
```

4) симметричное число конструируется так:

– перебираем цифры с 9 до 0 (цикл с параметром, изменяемым в обратном порядке – от “9” до “0”):

```
for c := '9' downto '0' do
```

– выводим цепочку таких цифр (цифра – текущее значение параметра цикла), длина которой равна половине всего количества таких цифр (если это количество нечетно, то дробная часть отбрасывается, то есть используется целочисленное деление):

Способ 1: с использованием стандартной функции	*Способ 2*: без использования стандартной функции
`write(StringOfChar(c, a[c] div 2));`	`for i := 1 to a[c] div 2 do write(c);`

– по завершении цикла перебора цифр от 9 до 0, если k = 1, то выводим одну цифру, количество которых было нечетно (она запомнена в переменной **c_odd**):

```
if k = 1 then write(c_odd);
```

– перебираем цифры с 0 до 9:

```
for c := '0' to '9' do
```

– выводим цепочку таких цифр (цифра – текущее значение параметра цикла), длина которой равна половине всего количества таких цифр (если это количество нечетно, то дробная часть отбрасывается, то есть используется целочисленное деление):

Способ 1: с использованием стандартной функции	*Способ 2*: без использования стандартной функции
`write(StringOfChar(c, a[c] div 2));`	`for i := 1 to a[c] div 2 do write(c);`

Полный текст программы:

```
program C4;
var a: array['0'..'9'] of integer;
    c, c_odd: char;
    i, k, s: integer;
```

```
begin
  for c:='0' to '9' do a[c]:=0;  // обнуляется массив-счетчик

  read(c);              // чтение символов из строки до точки
  while c<>'.' do
   begin
    if c in ['0' .. '9'] then a[c] := a[c] + 1;
    // если считанный символ – цифра, то соответствующий элемент
       массива-счетчика увеличивается на 1
    read(c);
  end;

// подсчет количества цифр, встречающихся нечетное число раз
  k := 0; s := 0;           // начальные обнуления
  for c := '0' to '9' do    // просмотр каждой цифры
   begin
    if a[c] mod 2 = 1 then  // если ее количество нечетно, то
     begin
      k := k + 1;           // увеличить счетчик
      c_odd := c            // и запомнить эту цифру
     end;
    s := s+a[c];
    // в любом случае подсчет суммарного количества всех цифр
   end;

  // проверка первого условия (единственный нуль)
  if (a['0'] = 1) and (s = 1) then
   begin
    writeln('YES');
    writeln('0');
   end else

  // иначе проверка второго условия (не более одной цифры встречается
     нечетное число раз, притом нули – не единственные цифры)
  if (s <> a['0']) and ((k = 0) or (k = 1)) then
   begin
    writeln('YES');

    // если условие соблюдается, то конструируется число
    for c := '9' downto '0' do
      write(StringOfChar(c,a[c] div 2));  // первая половина
    if k = 1 then write(c_odd);           // средний символ
    for c := '0' to '9' do
      write(StringOfChar(c,a[c] div 2));  // вторая половина

   // иначе ответ – 'NO'
   end else writeln('NO');
end.
```

Задача 4[6]. В командных олимпиадах по программированию для решения предлагается не больше 11 задач. Команда может решать предложенные задачи в любом порядке. Подготовленные решения команда посылает в единую проверяющую систему соревнований.

Вам предлагается написать эффективную, в том числе по используемой памяти, программу, которая будет статистически обрабатывать пришедшие запросы, чтобы определить

[6] С4/ДЕМО2012.

наиболее популярные задачи. Следует учитывать, что количество запросов в списке может быть очень велико, так как многие соревнования проходят с использованием Интернет.

Перед текстом программы кратко опишите используемый вами алгоритм решения задачи.

На вход программе в первой строке подаётся количество пришедших запросов *N*. В каждой из последующих *N* строк записано название задачи в виде текстовой строки. Длина строки не превосходит 100 символов, название может содержать буквы, цифры, пробелы и знаки препинания.

Пример входных данных:

```
6
A+B
Крестики-Нолики
Прямоугольник
Простой делитель
A+B
Простой делитель
```

Программа должна вывести список из трёх наиболее популярных задач с указанием количества запросов по ним. Если в запросах упоминаются менее трех задач, то выведите информацию об имеющихся задачах. Если несколько задач имеют ту же частоту встречаемости, что и третья по частоте встречаемости задача, их тоже нужно вывести.

Пример выходных данных для приведённого выше примера входных данных:

```
A+B 2
Простой делитель 2
Крестики-Нолики 1
Прямоугольник 1
```

Решение

1. Требуется подсчет количества каждой из задач. Всего задач может быть 11 видов. Следовательно, необходим массив-счетчик на 11 ячеек. Поскольку названия задач не могут являться индексами массива-счетчика, требуется также индексный массив из 11 ячеек, который должен хранить названия этих задач.

```
Var Count: array[1..11] of integer;  // массив-счетчик
    Names: array[1..11] of string;   // индексный массив
```

Однако есть дополнительная сложность: названия задач, которые нужно запоминать, и даже их реальное количество не известны. Поэтому требуется *динамически формировать индексный массив*.

2. Входные данные: вначале отдельно считывается количество строк, а затем в цикле с параметром вводятся строки – названия задач.

3. Обработка входных данных:

1) в индексном массиве путем просмотра с его начала количества заполненных его элементов (это количество изначально равно нулю) ищется элемент, равный считанному названию задачи;

```
Num:=0;
// количество заполненных элементов индексного массива первоначально
   равно нулю
. . .
  j:=1; // просмотр индексного массива с начала
  while (j<=Num) and (s<>Names[j]) do j:=j+1;
  // переходим к следующему элементу индексного массива, пока или не
     будут просмотрены все его заполненные элементы, или в нем не будет
     найдено считанное название задачи
```

2) если такой элемент найден, то элемент массива-счетчика с соответствующим индексом (j) увеличивается на 1;

```
  if j<=Num then  // значит, просмотр завершен раньше, чем просмотрены
                     все заполненные элементы, то есть в ячейке
                     с индексом j найдено требуемое название задачи
    Count[j]:=Count[j]+1
```

3) иначе (если такой элемент не найден) индексный массив наращивается на один элемент:

```
  else begin   // иначе
```

– в текущую (незанятую) ячейку индексного массива с индексом j записывается считанное новое название задачи;

```
    Names[j]:=s;
```

– в соответствующую ячейку (с индексом j) массива-счетчика записывается единица (одна такая задача уже встречена);

```
    Count[j]:=1;
```

– количество заполненных элементов индексного массива увеличивается на 1.

```
    Num:=Num+1
  end
```

Результат: заполненные по всем встреченным задачам индексный массив с названиями задач и массив-счетчик с их количествами.

4. Обработка массива-счетчика: требуется определить три вида задач, для которых количества будут наибольшими. Для этого проще всего отсортировать по убыванию массив-счетчик (и синхронно с ним индексный массив). Используется метод «пузырька» (попарное сравнение элементов и при необходимости их обмен местами); для упрощения решения задачи – с полным просмотром (без контроля досрочного прерывания сортировки, если массив уже отсортирован).

```
for i:=Num downto 2 do
  for j:=2 to i do
    if Count[j-1]<Count[j] then
     begin
      t:=Count[j]; Count[j]:=Count[j-1]; Count[j-1]:=t;
      s:=Names[j]; Names[j]:=Names[j-1]; Names[j-1]:=s;
     end;
```

ВАЖНО! Нужно не забывать, что при перестановке элементов в массиве-счетчике нужно одновременно переставлять соответствующие им элементы индексного массива, чтобы сохранить соответствие этих массивов.

5. Вывод данных:

1) если найдено только три вида задач или меньше, то надо выводить информацию обо всех них (тогда в переменной `j` запоминается номер последней запомненной задачи), иначе (если видов задач больше) – только о трех первых видах (тогда в переменной `j` запоминается номер 3);

2) массив-счетчик просматривается с начала и до тех пор, пока не исчерпано количество заполненных его элементов (`Num`) и пока количество задач текущего вида не меньше количества задач того вида, что в отсортированных массивах (счетчике и индексном) расположен на месте под номером `j`. Очевидно, что при этом если различных задач меньше трех, то будет выведена информация обо всех них, а если их больше трех, то будет выведена информация о трех первых по количеству видах задач, а также о следующих задачах (четвертой и т. д.), если их количество совпадает с количеством задач третьего вида.

```
if Num >= 3 then j := 3 else j := Num;
i := 1;
while (i <= Num) and (Count[i] >= Count[j]) do
 begin
  WriteLn(Names[i], ' ', Count[i]);
  i := i + 1;
 end
```

Полный текст программы:

```
Program C4;
Var Count: array[1..11] of integer;  // массив-счетчик
    Names: array[1..11] of string;   // индексный массив
    n, Num, i, j, t: integer;
    s: string;
Begin
  Num:=0;
  // изначально в индексном массиве не заполнено ни одного элемента
  ReadLn(N);         // считано количество строк
  for i:=1 to N do  // чтение строк входных данных
   begin
    ReadLn(S);       // считано очередное название задачи

    // считанное название задачи ищется в индексном массиве:
    j:=1;
    while (j<=Num) and (s<>Names[j]) do j:=j+1;
    if j<=Num then           // если задача найдена в j-й ячейке,
      Count[j]:=Count[j]+1  // то увеличиваем соответствующий счетчик
   else begin   // иначе название задачи добавляется в конце индексного
                  массива (в ячейку с индексом j)
      Names[j]:=s;
      Count[j]:=1;
      Num:=Num+1
    end
   end;

  // сортировка массива-счетчика по убыванию и синхронно с ним – индексного
    массива (методом «пузырька»)
  for i:=Num downto 2 do
    for j:=2 to i do
      if Count[j-1]<Count[j] then
       begin
        t:=Count[j]; Count[j]:=Count[j-1]; Count[j-1]:=t;
        s:=Names[j]; Names[j]:=Names[j-1]; Names[j-1]:=s;
       end;
```

```
  // вывод результата
  if Num >= 3 then j := 3 else j := Num;
  i := 1;
  while (i <= Num) and (Count[i] >= Count[j]) do
   begin
    WriteLn(Names[i], ' ', Count[i]);
    i := i + 1;
   end
end.
```

ЗАДАЧИ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

Задача 1[7]. После единых выпускных экзаменов по информатике в район пришла информация о том, какой ученик какой школы сколько баллов набрал.

Районный методист решила выяснить номер школы, ученики которой набрали наибольший средний балл, с точностью до целых.

Программа должна вывести на экран номер такой школы и её средний балл.

Если наибольший средний балл набрало больше одной школы – вывести количество таких школ.

Напишите эффективную, в том числе и по используемой памяти, программу (укажите используемую версию языка программирования, например Borland Pascal 7.0), которая должна вывести на экран требуемую информацию. Известно, что информатику сдавало больше 5-ти учеников района. Также известно, что в районе школы с некоторыми номерами не существуют.

На вход программе сначала подаётся число учеников, сдававших экзамен. В каждой из следующих *N* строк находится информация об учениках в формате:

<Фамилия> <Имя> <Номер школы> <Количество баллов>

где **<Фамилия>** – строка, состоящая не более чем из 30 символов без пробелов, **<Имя>** – строка, состоящая не более чем из 20 символов без пробелов, **<Номер школы>** – целое число в диапазоне от 1 до 99, **<Количество баллов>** – целое число в диапазоне от 1 до 100. Эти данные записаны через пробел, причём ровно один между каждой парой (то есть всего по три пробела в каждой строке).

Пример входной строки:

```
Иванов Иван 50 87
```

Пример выходных данных:

```
50 74
```

Другой вариант выходных данных:

```
7
```

Задача 2[8]. После единых выпускных экзаменов по информатике в район пришла информация о том, какой ученик какой школы сколько баллов набрал.

Районный методист решила выяснить фамилии учеников, которые набрали наибольший балл, по каждой школе в отдельности, но только если из школы информатику сдавало не меньше 3 человек. Если в школе информатику сдавало меньше 3 человек, информацию по этой школе выводить не нужно.

Программа должна вывести на экран информацию в виде: **<Номер школы> <Фамилия ученика>** в отдельной строке для каждой школы.

---

[7] С4/Вариант 2 [1, с. 70].

[8] С4/Вариант 4 [1, с. 107].

Напишите эффективную, в том числе и по используемой памяти, программу (укажите используемую версию языка программирования, например Borland Pascal 7.0), которая должна вывести на экран требуемую информацию. Известно, что информатику сдавало больше 5-ти учеников района. Также известно, что в районе школы с некоторыми номерами не существуют.

На вход программе сначала подаётся число учеников, сдававших экзамен. В каждой из следующих *N* строк находится информация об учениках в формате:

<Фамилия> <Имя> <Номер школы> <Количество баллов>,

где **<Фамилия>** – строка, состоящая не более чем из 30 символов без пробелов, **<Имя>** – строка, состоящая не более чем из 20 символов без пробелов, **<Номер школы>** – целое число в диапазоне от 1 до 99, **<Количество баллов>** – целое число в диапазоне от 0 до 100. Эти данные записаны через пробел, причём ровно один между каждой парой (то есть всего по три пробела в каждой строке).

Пример входной строки:

```
Иванов Иван 50 87
```

Пример выходных данных:

```
5 Иванов
50 Петров
74 Сидоров
```

ОТВЕТЫ ДЛЯ САМОПРОВЕРКИ

Задача 1.

Решение

1. Нужно определять средние баллы учеников каждой школы. Для этого требуется отдельно подсчитывать суммы баллов учеников каждой школы и количества учеников этих школ. Номер школы – двузначный, то есть может быть равен от 1 до 99. Следовательно, нужно использовать два массива-счетчика с индексами от 1 до 99 для подсчета сумм и количеств (либо можно использовать один двумерный массив-счетчик).

```
var s, k: array[1..99] of integer;
```

В качестве индексов массива-счетчика используются сами считываемые номера школ. Кроме того, хотя в массивах-счетчиках могут использоваться не все ячейки, список школ заранее не известен, поэтому использовать индексный массив нерационально.

2. Входные данные: вначале считывается количество строк данных *N*, затем в цикле с параметром считываются сами строки данных.

При этом фамилия и имя ученика – это лишние данные, их нужно пропускать путем посимвольного чтения до завершающего пробела включительно.

Далее номер школы и количество баллов (целые числа) считываются и обрабатываются.

3. Обработка считываемых данных:

1) в первом массиве-счетчике значение элемента с индексом, равным считанному номеру школы, увеличивается на считанное значение баллов (подсчет суммы баллов всех учеников данной школы);

2) во втором массиве-счетчике значение элемента с индексом, равным считанному номеру школы, увеличивается на 1 (подсчет количества учеников данной школы).

4. Обработка массива-счетчика:

1) вычисление среднего балла по каждой школе путем целочисленного деления суммы баллов на количество учеников (деление целочисленное, так как по условию задачи требуется определять средние значения с точностью до целых; по этой же причине можно объявить массив-счетчик целого типа, а не вещественного);

2) поиск максимума с одновременным подсчетом количества элементов массива-счетчика, равных максимальному (типовой алгоритм, но переприсваиваются индексы, а сравниваются соответствующие элементы массивов).

5. Вывод результатов: если количество найденных элементов, равных максимальному, равно 1, то вывести номер школы (он равен найденному индексу максимального элемента – max) и средний балл для этой школы (**s[max]**). Иначе вывести количество элементов, равных максимальному (**nmax**).

Полный текст программы:

```
program C4;
var s, k: array[1..99] of integer;  // массивы-счетчики
    ch: char;
    i, N, sh, ball, max, nmax: integer;
begin
  for i:=1 to 99 do  // обнуление массивов-счетчиков
   begin
    s[i]:=0; k[i]:=0
   end;

  readln(N);  // считали количество строк
  for i:=1 to N do  // считывание строк входных данных
   begin
    repeat
      read(ch)
    until ch=' ';  // пропуск фамилии
    repeat
      read(ch)
    until ch=' ';  // пропуск имени
    readln(sh,ball);  // чтение номера школы и балла
    s[sh]:=s[sh]+ball;  // подсчет суммы баллов для данной школы
    k[sh]:=k[sh]+1      // подсчет количества ее учеников
   end;

  // вычисление среднего балла для школ, для которых есть данные
  for i:=1 to 99 do
    if k[i]>0 then s[i]:=s[i] div k[i];

  // поиск максимума и количества элементов, равных максимуму
  max:=1; nmax:=1;
  // первоначально приняли за максимальный первый элемент,
  // счетчик количества таких элементов равен 1
  for i:=2 to 99 do  // просмотр остальных элементов
    if s[i]>s[max] then
    // если текущий элемент больше предполагаемого максимального, то
     begin
      max:=i;  // этот элемент будет новым максимумом
      nmax:=1  // а счетчик сбрасывается в 1
     end else
    if s[i]=s[max] then
    // иначе если текущий элемент равен максимуму, то
      nmax:=nmax+1;  // счетчик таких элементов увеличиваем на 1
  end;

  // вывод результатов
  if nmax=1 then writeln(max,' ', s[max])  // для одной школы
  else writeln (nmax);                     // для нескольких школ
end.
```

Задача 2.

Решение

1. Нужно запоминать для каждой школы:

– максимальный балл учеников;

– имя ученика с максимальным баллом;

– общее количество учеников данной школы.

Для этого требуется три отдельных массива: два массива-счетчика и индексный массив, заполняемый по мере обработки данных. Номер школы – двузначный, то есть может быть равен от 1 до 99. Следовательно, диапазон индексов этих массивов-счетчиков – от 1 до 99.

```
var num, bal: array[1..99] of integer;  // массивы-счетчики
    name: array[1..99] of string[52];   // индексный массив
```

В качестве индексов массива-счетчика используются сами считываемые номера школ.

2. Входные данные: вначале считывается количество строк данных *N*, затем в цикле с параметром считываются сами строки данных.

При этом фамилия ученика важна и посимвольно считывается в отдельную переменную – «буфер» до завершающего пробела включительно.

Имя ученика – это лишнее данное, оно пропускается путем посимвольного чтения до завершающего пробела включительно.

Далее номер школы и количество баллов (целые числа) считываются и обрабатываются.

3. Обработка считываемых данных:

1) проверяется: если считанное значение баллов больше, чем запомненное ранее для данной школы в элементе массива-счетчика **bal** с индексом, равным считанному номеру школы, то:

– в этот элемент массива-счетчика **bal** записывается новое максимальное значение;

– в элемент массива name с индексом, равным номеру школы, записывается фамилия текущего ученика как предположительно набравшего максимальный балл в данной школе;

2) в массиве-счетчике num значение элемента с индексом, равным считанному номеру школы, увеличивается на 1 (подсчет количества учеников данной школы).

4. Обработка массива-счетчика после завершения обработки входных данных не требуется.

5. Вывод результатов:

1) просматриваются все номера школ от 1 до 99;

2) если для данной школы количество учеников больше или равно трем, то выводится номер школы (параметр цикла), а затем запомненная фамилия ученика для данной школы (из индексного массива).

Полный текст программы:

```
program C4;
var num, bal: array[1..99] of integer;  // массивы-счетчики
    name: array[1..99] of string[52];   // индексный массив
    s: string[52];  // «буферная» переменная для фамилии
    ch: char;
    i, N, sh, ball: integer;
begin
  for i:=1 to 99 do  // инициализация массивов-счетчиков
   begin
    num[i]:=0;  // количества учеников - обнуляются
    bal[i]:=-1  // в качестве первого предполагаемого максимума берется
          значение, заведомо меньшее минимально возможного числа баллов
   end;
```

```
  readln(N);  // считано количество строк
  for i:=1 to N do  // чтение строк входных данных
   begin
    s: = ' '; // «обнуление» строки для чтения фамилии
    repeat
      read(ch);
      s := s+ch
    until ch=' '; // считана фамилия
    repeat
      read(ch)
    until ch=' '; // пропущено имя
    readln(sh,ball);  // считаны номер школы и балл

    if ball > bal[sh] then
    // если считанный балл больше, чем ранее запомненный максимум
       для данной школы, то
     begin
      bal[sh]:= ball; // запоминаем этот балл как новый максимум
                      для данной школы
      name[sh]:=s     // запоминаем фамилию ученика с этим баллом
     end;

    num[sh]:=num[sh]+1  // подсчет количества учеников данной школы
   end;

  // вывод результатов
  for i:=1 to 99 do
    if num[i]>=3 then  // если в данной школе сдавало экзамен не менее
                       трех учеников, то
      writeln(i,' ',name[i]);  // вывести номер школы и фамилию ученика,
                                  запомненную для нее
end.
```

Богомолова Ольга Борисовна,
доктор педагогических наук,
почетный работник сферы образования Российской Федерации,
Заслуженный учитель города Москвы, учитель информатики и математики ГБОУ СОШ № 1360, г. Москва,

Усенков Дмитрий Юрьевич,
ГБОУ СОШ № 1360, г. Москва.