Практическая работа по информатике №ED27

Памятка по выполнению:

1. Изучите поставленную задачу. Изучите прилагаемый пример решения.
2. Закройте пример решения, не подглядывая туда, и напишите самостоятельно программу для решения задачи. Отладьте её на прилаеаемых примерах + составленных самостоятельно.
3. Работа выполняется сначала на компьютере, потом программа обязательно переписывается в тетрадь. Это вызвано требованиями ЕГЭ-2017 - программа сдаётся на бланке!
4. Текст программы должен быть обязательно отформатирован отступами.
5. В тексте программы обязательно должны быть комментарии, поясняющие текущие действия и алгоритм решения. Количество комментариев должно быть осмысленным - каждый чих комментировать не нужно, но в ключевых местах комментарии должны быть. В частности, обязательно комментируется предназначение каждой переменной при её объявлении.
6. Задание оценивается в обязательном порядке только для тех, кто сдаёт ЕГЭ по предмету. Остальным выставляются только положительные оценки (по желанию).
7. В варианте ЕГЭ программа должна читать данные с клавиатуры, но проверять её так совершенно неудобно, поэтому мы делаем чтение из файла, а потом переделываем в вариант для клавиатуры. Представлено должно быть 2 электронные версии программы - с файлом и с клавиатуры, в тетради - только для клавиатуры.

Задача 1. После экзаменов по информатике в район пришла информация о том, какой ученик какой школы сколько баллов набрал. Эта информация в том же виде была разослана в школы в виде текстового файла "results.txt".

Завуч школы № 1 решила наградить печеньками двух учащихся, которые лучше всех в школе сдали экзамен. На большее количество наград печенек не хватило.

Программа должна прочитать данные из файла и вывести на экран фамилии и имена этих учеников. При этом:

- ➔ Если наибольший балл набрало больше 2 человек - нужно вывести количество таких человек (награждение невозможно);
- ➔ Если наибольший балл набрал один человек, а следующий балл набрало несколько человек - нужно вывести только фамилию и имя лучшего (будет награждён 1 участник);
- ➔ Если учеников школы №1 в файле не оказалось, программа должна об этом сообщить.

Напишите эффективную, в том числе и по используемой памяти, программу (укажите используемую версию языка программирования: **PascalABC.NET 3.2**), которая должна вывесли на экран требуемую информацию.

На вход программы подаётся текстовый файл "results.txt", лежащий в той же папке, что и программа. Создайте такой файл в приложении "Блокнот" самостоятельно и отладьте свою программу максимально подробно.

Первой строкой в файле записано целое число N - количество записей. В каждой из последующих N строках находится информация об учениках в формате:

<Фамилия>(пробел) <Имя>(пробел) <Номер школы>(пробел) <Колич.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 1..99, <Количество баллов> - целое число в диапазоне 1..100. Эти данные записаны через пробел, причём ровно один между каждой парой (т.е. всего по 3 пробела в каждой строке). Пример входной строки:

Иванов Петя 8 35

Пример выходных данных (вариант 1):

Белов Александр

Смолин Алексей

Пример выходных данных (вариант 2):

3

Пример выходных данных (вариант 3):

Анчуткин Иван

Пример выходных данных (вариант 4):

Нет учеников нашей школы

Комментарии к задаче:

1) Запомните версию языка программирования: **PascalABC.NET 3.2**
2) Если написано "эффективная по памяти программа", это означает, что данные нельзя запоминать в массив, а нужно обрабатывать сразу, по мере поступления входных строк. Продумайте несложный вариант алгоритма, который запоминает не только максимум, но и второй по величине результат. Принцип тот же.
3) В алгоритме будет много логических ловушек, связанных с тем, что количества баллов могут быть совершенно любыми - например, все могут получить один и тот же балл, баллы могут идти в любом порядке и т.п.
4) Чтобы работать с файлами, нужно:
   - ➔ Завести файловую переменную в разделе `var f: text;`
   - ➔ В самом начале программы открыть файл вызовом процедуры `Reset(f,'results.txt');`
   - ➔ Использовать функции чтения в модификации `readln(f, <список переменных>);`
   - ➔ В конце программы закрыть файл вызовом процедуры `Close(f);`

Решение.

```
// PascalABC.NET 3.2
program ege27_1;
var
   ch: char;  // один символ для чтения входной строки
   i: integer;
   n: integer; // число записей в файле
   n_spaces: integer; // подсчёт пробелов при вводе
   scool,mark: integer; // номер школы и оценка текущей записи
   max1,max2: integer;  // текущие оценки для 1 и 2 места
   nmax1,nmax2: integer; // текущие: счётчики количества встреченных одинаковых баллов для 1 и 2 места
   fam,fmax1,fmax2: string;  // текущие: фамилия, фамилия 1 места и 2 места.
   f: text; // ссылка на входной файл
begin
   max1:=0; // По условию баллы 1..100, поэтому первый же балл должен обновить это значение.
  // Если же останется 0, то это значит, что интересующих нас записей вообще встречено не было.
  // Обратим внимание, что max2 инициализировать не обязательно, т.к. алгоритм сделает это сам, если только попадётся хоть один ученик нашей школы.
   Reset(f,'results_9.txt'); // Открываем файл входных данных
  // Читаем количество записей:
   readln(f,n);
   // Читаем n записей из файла:
   for i:=1 to n do
  begin
    // Читаем фамилию и имя ученика:
    fam:='';
    n_spaces:=0;
    repeat
      read(f,ch);
      if ch=' ' then n_spaces:=n_spaces+1;
      if n_spaces<2 then fam:=fam+ch;
    until n_spaces=2;
    read(f,scool);  // Читаем номер школы. Внимание: ln - не надо!
    readln(f,mark); // Читаем оценку. Внимание: ln - надо!
   // Анализируем полученные результаты для данного ученика:
    if scool<>1 then continue; // если школа не 1,то переход сразу к следующему i
   // Здесь обрабатываются только ученики нашей школы:
    if mark>max1 then // встретили чемпионскую отметку
    begin
      max2:=max1; fmax2:=fmax1; nmax2:=nmax1; // передвинем бывшего чемпиона на 2 место
      max1:=mark; fmax1:=fam;   nmax1:=1;     // пока что это 1 такой высокий балл в нашей истории, не забудем отметить.
    end else if mark=max1 then     // встретили такую же оценку, как у чемпиона
    begin
      nmax1:=nmax1+1;
      fmax2:=fam;    // В случае 2-х чемпионов фамилию второго сохраним здесь
    end else if mark>max2 then // встретили балл, лучший чем 2 место
    begin
      max2:=mark; fmax2:=fam; nmax2:=1;
    end else if mark=max2 then nmax2:=nmax2+1;
  end; // следующая строка
  // Выводим результаты:
   if max1=0 then writeln('Учеников нашей школы не обнаружено') // Если вообще не было строк с нашей школой
   else // Были наши, нужно смотреть на счётчики:
   if (nmax1=2) or (nmax1=1) and (nmax2=1) then begin writeln(fmax1); writeln(fmax2); end // можем наградить двоих (2 случая)
   else if (nmax1=1) and (nmax2<>1) then writeln(fmax1)  // вторых мест или нет (0), или больше одного, поэтому награждаем только 1 место
   else writeln(nmax1); // никого наградить не можем, т.к. наград не хватает
   Close(f); // закроем файл
end.
```

Задача 2. После экзаменов в район пришла информация о том, какой ученик какой школы сколько баллов набрал.

Районный методист решил 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 т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна вывести на экран требуемую информацию. Известно, что экзамен сдавало больше 5-ти учеников района. Также известно, что школы с некоторыми номерами не существуют.

На вход программы подаётся текстовый файл "results.txt", лежащий в той же папке, что и программа. Создайте такой файл в приложении "Блокнот" самостоятельно и отладьте свою программу максимально подробно.

Первой строкой в файле записано целое число N - количество записей. В каждой из последующих N строках находится информация об учениках в формате:

<Фамилия>(пробел) <Имя>(пробел) <Номер школы>(пробел) <Колич.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 1..99, <Количество баллов> - целое число в диапазоне 1..100. Эти данные записаны через пробел, причём ровно один между каждой парой (т.е. всего по 3 пробела в каждой строке). Пример входной строки:

Иванов Петя 50 87

Пример выходных данных:

50 74

Другой вариант выходных данных:

7

Подсказка: Входной поток запоминать в массивы не нужно (эффективность по памяти). Но хранить информацию по каждой школе в массиве придётся.

```
// PascalABC.NET 3.2
 program ege27_2;
var
   n,i,p,nsh,mark,mean,max,nmax,num: integer;
   line: string;
   table_sum,table_n: array[1..99] of integer;
    f: text;
begin
   for i:=1 to 99 do begin table_sum[i]:=0; table_n[i]:=0; end;
   Reset(f,'data_ege27_2.txt');
   readln(f,n);  writeln('В файле ожидается ',n,' записей');
   for i:=1 to n do
  begin
     Readln(f,line);
     p:=Pos(' ',line); p:=Pos(' ',line,p+1);
    nsh:=ReadIntegerFromString(line,p);
    mark:=ReadIntegerFromString(line,p);
    table_sum[nsh]:=table_sum[nsh]+mark;
     table_n[nsh]:=table_n[nsh]+1;
  end;
  // Ввод данных закончен, начинаем анализ по школам:
  max:=0;
  for i:=1 to 99 do if table_n[i]>0 then
 begin
    mean:=table_sum[i] div table_n[i];
    if mean>max then begin max:=mean; nmax:=1; num:=i; end
    else if mean=max then nmax:=nmax+1;
  end;
  if nmax>1 then writeln(nmax) else writeln(num,' ',max);
 Close(f);
end.
```

Задача 3. После экзаменов в район пришла информация о том, какой ученик какой школы сколько баллов набрал.

Районный методист решил выяснить номер школы, ученики которой набрали средний балл по школе, больший, чем районный средний балл (все средние баллы вычисляются с точностью до целых).

Программа должна вывести на экран номера таких школ, в любом порядке.

Если такая школа окажется только одна - вывести также средний балл по этой школе, с указанием, что это - средний балл.

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна вывести на экран требуемую информацию. Известно, что экзамен сдавало больше 5-ти учеников района. Также известно, что школы с некоторыми номерами не существуют.

На вход программы подаётся текстовый файл "results.txt", лежащий в той же папке, что и программа. Создайте такой файл в приложении "Блокнот" самостоятельно и отладьте свою программу максимально подробно.

Первой строкой в файле записано целое число N - количество записей. В каждой из последующих N строках находится информация об учениках в формате:

<Фамилия>(пробел) <Имя>(пробел) <Номер школы>(пробел) <Колич.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 1..99, <Количество баллов> - целое число в диапазоне 1..100. Эти данные записаны через пробел, причём ровно один между каждой парой (т.е. всего по 3 пробела в каждой строке). Пример входной строки:

Иванов Петя 50 87

Пример выходных данных:

5 50 74 87

Другой вариант выходных данных:

```
7
Средний балл = 74
```

```
// PascalABC.NET 3.2
 program ege27_3;
var
   n,i,p,nsh,mark: integer;
   raion,n_out: integer;
   line: string;
   s,k: array[1..99] of integer; // таблицы сумм/средних + число учеников для каждой
школы
   f: text;
begin
   for i:=1 to 99 do begin s[i]:=0; k[i]:=0; end;
   Reset(f,'data2.txt');
  readln(f,n);
   for i:=1 to n do
  begin
     Readln(f,line);
     p:=Pos(' ',line); p:=Pos(' ',line,p+1);
    nsh:=ReadIntegerFromString(line,p);
    mark:=ReadIntegerFromString(line,p);
     s[nsh]:=s[nsh]+mark; // накапливаем сумму баллов
     k[nsh]:=k[nsh]+1;    // накапливаем количество сдавших учеников
   end;
   // Ввод данных закончен, начинаем анализ по школам:
  // Вычисляем средние баллы для каждой школы и по району:
   raion:=0; // сначала это сумма баллов по району
   for i:=1 to 99 do if k[i]>0 then
  begin
     raion:=raion+s[i];
     s[i]:=s[i] div k[i]; // теперь s[i] - средний балл по данной школе
   end;
   raion:=raion div n; // это средний балл по району. Кол-во учеников и так известно!
  // Просто выводим номера школ, для которых есть данные и средний балл по
  // этой школе превышает средний балл по району
   n_out:=0;
   for i:=1 to 99 do if k[i]>0 then
  begin
     if s[i]>raion then
    begin
       write(i,' ');
       mark:=s[i]; // запомним средний балл для школы на случай, если она одна
       n_out:=n_out+1; // считаем количество выведенных школ
     end;
   end;
   // Выведем дополнительную информацию, если нужно:
   if n_out=1 then
  begin
     writeln; // Перейдём на новую строку
     writeln('Средний балл = ',mark);
   end
   else if n_out=0 then writeln('Ни одной школы, удовлетворяющей условию,не найдено');
  Close(f);
end.
```

Задача 4. После единых выпускных экзаменов по информатике в район пришла информация о том, какой ученик какой школы сколько баллов набрал.

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

Внимание! Здесь пример некорректной постановки задачи на ЕГЭ! В условии не сказано, что делать, если в этой школе максимальный балл набрало несколько человек. Решение задачи подразумевает дополнительное условие: такого быть не может. Но явно это не сказано, а и быть такое может запросто. Попытка решить эту задачу при допущении множества получивших одинаковый балл резко усложнит программу. Поэтому мы дополним условие задачи следующим абзацем:

Если оказалось, что максимальный балл в данной школе набрало несколько человек, то вывести об этом сообщение с указанием этого количества набравших максимальный балл. Фамилии в этом случае выводить не нужно.

Программа должна вывести на экран информацию в виде:

1) <№ школы> <Фамилия чемпиона>
2) <№ школы> <Вывести навозможно: > <Количество чемпионов >

в отдельной строке для каждой школы.

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна вывести на экран требуемую информацию. Известно, что экзамен сдавало больше 5-ти учеников района. Также известно, что школы с некоторыми номерами не существуют.

На вход программы подаётся текстовый файл "results.txt", лежащий в той же папке, что и программа. Создайте такой файл в приложении "Блокнот" самостоятельно и отладьте свою программу максимально подробно.

Первой строкой в файле записано целое число N - количество записей. В каждой из последующих N строках находится информация об учениках в формате:

<Фамилия>(пробел) <Имя>(пробел) <Номер школы>(пробел) <Колич.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 1..99, <Количество баллов> - целое число в диапазоне 0..100. Эти данные записаны через пробел, причём ровно один между каждой парой (т.е. всего по 3 пробела в каждой строке). Пример входной строки:

```
Иванов Петя 50 87
```

Пример выходных данных:

```
5 Иванов
50 Петров
75 Сидоров
80 Вывести невозможно: 2 чемпиона
81 Задрипищев
```

Решение.

```
// PascalABC.NET 3.2
 program ege27_4;
var
   n,i,p1,p2,nsh,mark: integer;
   line: string; // строка для ввода данных

   max:  array[1..99] of integer; // максимальный балл для каждой школы
   nmax: array[1..99] of integer; // количество чемпионов для каждой школы
   k:    array[1..99] of integer; // количества учеников для каждой школы
   fam:  array[1..99] of string; // имя чемпиона для каждой школы

   f: text; // ссылка на файл
begin
   for i:=1 to 99 do begin max[i]:=-1; k[i]:=0; end;
   Reset(f,'data3.txt');
  readln(f,n);
   for i:=1 to n do
  begin
     Readln(f,line);
     p1:=Pos(' ',line); p1:=Pos(' ',line,p1+1);
    p2:=p1;
    nsh:=ReadIntegerFromString(line,p2);
    mark:=ReadIntegerFromString(line,p2);
     k[nsh]:=k[nsh]+1;    // накапливаем количество сдавших учеников
     if mark>max[nsh] then
    begin
       max[nsh]:=mark;
       fam[nsh]:=Copy(line,1,p1-1);
       nmax[nsh]:=1;
     end else if mark=max[nsh] then Inc(nmax[nsh]);
   end;
   // Выводим результаты:
   for i:=1 to 99 do if k[i]>3 then
   if nmax[i]=1 then writeln(i,' ',fam[i])
   else writeln(i,' Вывести невозможно: ',nmax[i],' чемпионов');
  Close(f);
end.
```

Задача 5. После единых выпускных экзаменов по информатике в район пришла информация о том, какой ученик какой школы сколько баллов набрал.

В районе считается подозрительной ситуация, когда в школе более двух учащихся набирают одинаковый наибольший балл по школе.

Районный методист решил выяснить номера таких школ.

Программа должна вывести номера таких школ, в любом порядке.

Если такая школа окажется одна, нужно вывести наибольший балл в этой школе с указанием, что это - наибольший балл.

Если таких школ не окажется, нужно вывести об этом сообщение.

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна вывести на экран требуемую информацию. Известно, что экзамен сдавало больше 5-ти учеников района. Также известно, что школы с некоторыми номерами не существуют.

На вход программы подаётся текстовый файл "results.txt", лежащий в той же папке, что и программа. Создайте такой файл в приложении "Блокнот" самостоятельно и отладьте свою программу максимально подробно.

Первой строкой в файле записано целое число N - количество записей. В каждой из последующих N строках находится информация об учениках в формате:

<Фамилия>(пробел) <Имя>(пробел) <Номер школы>(пробел) <Колич.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 1..99, <Количество баллов> - целое число в диапазоне 0..100. Эти данные записаны через пробел, причём ровно один между каждой парой (т.е. всего по 3 пробела в каждой строке).

Пример входной строки:

```
Иванов Петя 50 87
```

Пример выходных данных:

```
5 50 74 87
```

Другой вариант выходных данных:

```
7
Наибольший балл = 74
```

Третий вариант выходных данных:

```
Нет таких школ
```

Решение.

```
// PascalABC.NET 3.2
 program ege27_5;
var
   n,i,p,nsh,mark,n_out,imax: integer;
   line: string; // строка для ввода данных

   max:  array[1..99] of integer; // максимальный балл для каждой школы
   nmax: array[1..99] of integer; // количество чемпионов для каждой школы

   f: text; // ссылка на файл
begin
   for i:=1 to 99 do max[i]:=-1;
   Reset(f,'data6.txt');
  readln(f,n);
   for i:=1 to n do
  begin
     Readln(f,line);
     p:=Pos(' ',line); p:=Pos(' ',line,p+1);
    nsh:=ReadIntegerFromString(line,p);
    mark:=ReadIntegerFromString(line,p);
     if mark>max[nsh] then
    begin
       max[nsh]:=mark;
       nmax[nsh]:=1;
     end else if mark=max[nsh] then Inc(nmax[nsh]);
   end;
   // Выводим результаты:
   n_out:=0;
   for i:=1 to 99 do if (max[i]>0) and (nmax[i]>2) then
  begin
     write(i,' ');
    imax:=i;
    Inc(n_out);
   end;
   if n_out=1 then writeln('Наибольший балл = ',max[imax]);
   if n_out=0 then writeln('Нет таких школ');
  Close(f);
end.
```

Задача 6. При программировании школьной тестирующей системы по английскому языку выяснилось, что файлы с вопросами к тестам легко доступны, и каждый может перед тестом открыть их и заранее узнать вопросы. Было решено закодировать файлы. Для этого придумали следующий алгоритм.

Каждая строка файла кодируется отдельно.

В каждой строке ищутся отдельные слова, и все символы слова сдвигаются по алфавиту циклически вправо на длину слова.

Словом считается любая последовательность подряд идущих символов латинского алфавита, строчных и прописных.

Циклический сдвиг символа по алфавиту вправо на X - замена символа на символ, стоящий в алфавите на X позиций дальше. Если при этом происходит выход за пределы алфавита, счёт начинается с начала алфавита.

Пример циклического сдвига символов нв 3 позиции:

- буква "E" превращается в букву "H"
- буква "t" превращается в букву "w"
- буква "Y" превращается в букву "B"

Напишите эффективную, в т.ч. по используемой памяти, программу (кажите версию языка программирования), которая должна закодировать строку по указанному алгоритму.

На вход программе подаётся строка, состоящая из не более чем 250 символов латинского алфавита, пробелов, знаков препинания, разного рода скобок, кавычек и других символов. Строка заканчивается символом #. Других символов # в строке нет.

Программа должна вывести закодированную по указанному алгоритму строку.

Пример входных данных:

```
Day, mice. "Year" - a mistake#
```

Пример выходных данных:

```
Gdb, gmgi. "Ciev" - b tpzahrl#
```

Подсказка. Что нужно знать для решения данной задачи:

- В английском алфавите 26 букв
- Буквы в кодировке ASCII расположены строго по алфавиту. Сначала идёт блок заглавных букв A...Z, потом, с некоторым смещением, блок строчных букв a...z:

ASCII Code Chart

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

- Какая именно в системе кодировка, нас не особенно волнует, так как в Unicode английские символы расположены относительно друг друга так же, как в таблице ASCII
- Код символа в Паскале можно получить функцией `код = Ord(ch)`, где ch - символ (char)
- Обратное преобразование делается функцией `ch = Chr(код)`
- Для символьных переменных определены операции типа больше/меньше, в смысле их следования в алфавитном порядке.
- Для упрощения проверок при учёте того факта, что одна и та же буква может быть как строчной, так и прописной, удобно использовать функции `UpCase(ch)/LowCase(ch)`
- Принадлежность символа ch к буквам может быть проверена условием

```
if (UpCase(ch)>='A') and (UpCase(ch)<='Z') then...
```

- Длину строки string s можно узнать функцией n = Length(s)

Решение.

```
// PascalABC.NET 3.2
 program ege27_6;
var
   s: string;
   flag: boolean;
   i,n,k,len: integer;
begin
   readln(s);
   n:=Length(s); // определим длину строки
   flag:=false; // флаг того,что мы обрабатываем текущее слово
   for i:=1 to n do // перебираем подряд все символы в строке от 1 до n
   begin
     if (UpCase(s[i])>='A') and (UpCase(s[i])<='Z') then // текущий символ - буква
алфавита?
       if flag=false then // если слово ещё не началось
       begin
         flag:=true; // значит, начнём слово
         len:=1; // длина этого слова пока 1 символ - тот,что считали сейчас
       end else len:=len+1 // слово уже идёт, просто подсчитываем его длину
     else // текущий символ оказался не буквой:
     if flag=true then // если этот символ завершил некое слово, то кодируем его
     begin
       flag:=false; // не забудем сбросить флаг
       for k:=1 to len do // для каждого символа этого найденного слова длиной len
       begin
         if Ord(UpCase(s[i-k]))-Ord('A')+len>25 then // выход за 26 букв англ.алфавита
           s[i-k]:=Chr(Ord(s[i-k])+len-26) // цикличность сдвига - вычтем число букв
         else
           s[i-k]:=Chr(Ord(s[i-k])+len); // цикличность не проявляется, просто сдвинем
       end;
     end;
   end;
  writeln(s);
end.
```

Задача 7. После единых выпускных экзаменов по информатике в район пришла информация о том, какой ученик какой школы сколько баллов набрал. По положению об экзамене каждый район сам определяет, за какой балл нужно поставить какую оценку.

Районный методист, использующий американский подход к оцениванию, решил, что оценку "5" должны получить 20% участников (целое число, с отбрасыванием дробной части).

Для этого нужно определить, какой балл должен был набрать ученик, чтобы получить "5".

Если невозможно определить такой балл, чтобы "5" получили ровно 20% участников, оценку "5" должны получить меньше участников, чем 20%.

Если таких участников не окажется (наибольший балл набрали более 20% участников) - эти и только эти ученики должны получить "5".

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна вывести на экран наименьший балл, который набрали участники, получившие "5". Известно, что экзамен сдавало больше 5-ти учеников. Также известно, что есть такое количество баллов, которое не получил ни один участник.

На вход программы подаётся текстовый файл "results.txt", лежащий в той же папке, что и программа. Создайте такой файл в приложении "Блокнот" самостоятельно и отладьте свою программу максимально подробно.

Первой строкой в файле записано целое число N - количество записей. В каждой из последующих N строках находится информация об учениках в формате:

<Фамилия>(пробел) <Имя>(пробел) <Номер школы>(пробел) <Колич.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 1..99, <Количество баллов> - целое число в диапазоне 0..100. Эти данные записаны через пробел, причём ровно один между каждой парой (т.е. всего по 3 пробела в каждой строке).

Пример входной строки:

```
Иванов Петя 50 87
```

Пример выходных данных:

78

Решение.

```
// PascalABC.NET 3.2
 program ege27_7;
var
   n,i,p,nsh,mark,n20,ball,sum: integer;
   line: string; // строка для ввода данных
   n_ball: array[0..100] of integer; // количества учеников, набравшие конкретный балл
   f: text; // ссылка на файл
begin
   for i:=0 to 100 do n_ball[i]:=0;
   Reset(f,'data1.txt');
  readln(f,n);
   for i:=1 to n do
  begin
     Readln(f,line);
     p:=Pos(' ',line); p:=Pos(' ',line,p+1);
    nsh:=ReadIntegerFromString(line,p);
    mark:=ReadIntegerFromString(line,p);
     Inc(n_ball[mark]); // подсчитываем количество человек для данного балла
   end;
   // Вычислим, сколько человек составляют наши 20%:
   n20:= n*20 div 100;
   // Подобираем такой балл, чтобы удовлетворить нашему условию:
   ball:=100; // это текущий "проходной балл" (на "отлично")
   sum:=n_ball[ball]; // это сумма получивших проходной и выше
   while sum<n20 do // уменьшаем проходной балл, покуда количество народу меньше 20%
   begin
     ball:=ball-1;
    sum:=sum+n_ball[ball];
   end;
   // Смотрим, что получилось:
  // 1) удачно совпало,что ровно 20% участников или 2) первый же балл дал превышение
20%:
   if (sum=n20) or (sum=n_ball[ball]) then writeln(ball) // тогда проходной получен
сразу
   else // 3) превышение 20% есть, но можно поднять планку, увеличив проходной балл:
   begin
    repeat
      ball:=ball+1; // поднимаем проходной балл
     until n_ball[ball]>0; // нужно найти такой балл, чтобы хоть кто-то его получил
     writeln(ball);
   end;
  Close(f);
end.
```

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

По положению об экзамене оценку «2» получают ученики, набравшие 40 и менее баллов. Оценку «3» получают 30% учеников среди оставшихся, за исключением тех из них, кто набрал больше 60 баллов. Величина 30% вычисляется целым делением с отбрасыванием дробной части.

Если количество «троечников» оказывается больше 30%, то следует выбрать меньшую границу для оценки «4» (но только если при этом «3» получит хоть кто-нибудь).

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна вывести на экран наибольший балл, который набрали участники, получившие "3" и количество таких учеников. Известно, что экзамен сдавало больше 50 учеников. Также известно, что есть такое количество баллов, которое не получил ни один участник.

На вход программы подаётся текстовый файл "results.txt", лежащий в той же папке, что и программа. Создайте такой файл в приложении "Блокнот" самостоятельно и отладьте свою программу максимально подробно.

Первой строкой в файле записано целое число N - количество записей. В каждой из последующих N строках находится информация об учениках в формате:

<Фамилия>(пробел) <Имя>(пробел) <Номер школы>(пробел) <Колич.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 1..99, <Количество баллов> - целое число в диапазоне 0..100. Эти данные записаны через пробел, причём ровно один между каждой парой (т.е. всего по 3 пробела в каждой строке).

Пример входной строки:

```
Иванов Петя 50 87
```

Пример выходных данных:

```
45 703
```

Примечания. В таких задачах требуется определить величины «граничных» баллов. Т.е. указать балл, набрав который, ученик получает ту или иную оценку. В данном случае для «2» это 40 баллов и ниже. Нужно подобрать такой балл (в диапазоне 41..60), получив который или менее (но не двойка) ученик получает «3».

Условие приведено «как есть» из ЕГЭ. В нём есть неоднозначно понимаемые моменты:

1) Оценку «3» получают 30% учеников среди оставшихся, за исключением тех из них, кто набрал больше 60 баллов.

Имеется ввиду следующее:

Оценку «3» получают 30% (или менее, но больше нельзя) от всех, кто не получил «2». При этом, если ученик набрал более 60 баллов, то он не должен получить «3». Т.е. верхнее значение границы для оценки «3» равно 60 баллов. Но граница может быть и ниже, так, чтобы выполнялось ограничение по 30%. Если, например, все, кроме двоечников, получили 61 балл и выше — число троек будет 0, несмотря на требование 30%.

2) Второе место, выделенное красным (но только если при этом «3» получит хоть кто-нибудь) означает следующее: указанный в ответе граничный балл должен быть реально получен хоть кем-то. Например, получены баллы ..30,31,40,41..., а граница 40 не подходит — её надо уменьшить. Нельзя указать граничный балл 39, его никто не получил. В этом случае граничный балл следует взять 31.

```
// PascalABC.NET 3.2
 program ege27_8;
var
   n,i,p,nsh,mark,n2,n30p,sum: integer;
   line: string; // строка для ввода данных
   k: array[0..100] of integer; // количества учеников, набравшие конкретный балл
   f: text; // ссылка на файл
begin
   for i:=0 to 100 do k[i]:=0;
   Reset(f,'data7.txt');
  readln(f,n);
   n2:=0; // количество двоечников
   for i:=1 to n do
  begin
     Readln(f,line);
     p:=Pos(' ',line); p:=Pos(' ',line,p+1);
    nsh:=ReadIntegerFromString(line,p);
    mark:=ReadIntegerFromString(line,p);
     Inc(k[mark]); // подсчитываем количество человек для данного балла
     if mark<=40 then Inc(n2); // заодно сразу считаем количество двоечников
   end;
   writeln('Двоек: ',n2);
   // Вычислим, сколько человек составляют наши 30%:
   n30p:= (n-n2)*3 div 10;
   writeln('Граница 30% = ',n30p);
   // Подобираем такой балл, чтобы удовлетворить нашему условию:
   i:=41; // это текущее значение границы для "3"
   sum:=k[i]; // это общее количество получивших "3" при данной величине границы
   while (sum<n30p) and (i<60) do // подбираем с учётом ограничений на величину
границы
   begin
     i:=i+1; // увеличиваем границу "4"
     sum:=sum+k[i];
   end;
   // Смотрим, что получилось:
   if sum=n30p then writeln(i,' ',sum) // если удачно ровно 30% - ответ готов
   else if (i=60) and (sum<n30p) then
  begin
     // подняли планку тройки до максимальной, но всё равно не набрали 30%
    // Вроде бы граничный балл 60, но может быть,его никто и не набрал:
     while k[i]=0 do i:=i-1; // ищем первый же набранный балл "вниз", если надо
     writeln(i,' ',sum);
   end
   else // значит, троечников получилось больше 30%:
   if sum=k[i] then writeln(i,' ',sum) // маловероятный, но возможный случай -
  // все троешники набрали одинаковый балл, разбить эту группу невозможно,
  // значит, оставляем как есть, пусть и больше 30%
   else
   begin  // перебрали количество, нужно найти меньший граничный балл:
     sum:=sum-k[i]; // от текущего балла уходим, значит, и минусуем всех, кто его
набрал
     repeat i:=i-1; until k[i]>0; // уменьшаем балл, пока не найдём такой, котоый хоть
кто-то получил
     writeln(i,' ',sum);
   end;
  Close(f);
end.
```

Задача 9. На вход программе подаётся последовательность символов, заканчивающихся символом #. Другие символы # во входной последовательности отсутствуют.

Программа должна вывести на экран латинскую букву, встречающуюся во входной последовательности наибольшее количество раз и число этих раз (во второй строке).

Если таких букв во входной последовательности окажется несколько, программа должна вывести на экран всех их, через пробел, в алфавитном порядке.

Строчные и прописные буквы не различаются.

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна решать поставленную задачу.

Пример 1:

Входные данные:

Day, mice. "Year" - a mistake#

Выходные даннные:

A

4

Пример 2:

Входные данные:

ABCD ABCE ABCF#

Выходные даннные:

A B C

3

Подсказки те же, что к задаче №6.

```
// PascalABC.NET 3.2
program ege27_9;
var
  ch: char;
  k: array[0..25] of integer; // 26 букв английского алфавита
  i,max: integer;

begin
  for i:=0 to 25 do k[i]:=0; // не забудем обнулить все счётчики!
  max:=0;
  repeat
    read(ch);
    if ch<>'#' then
    begin
      ch:=UpCase(ch); // теперь символ в любом случае в верхнем регистре
      // Но там могут быть не только буквы, но и другие символы!
      i:=Ord(ch)-Ord('A');
      if (i>=0) and (i<=25) then // если это буква алфавита 0..25
      begin
        Inc(k[i]); // увеличиваем счётчик попаданий для данной буквы
        if k[i]>max then max:=k[i]; // заодно ищем максимальную частоту встречаемости
      end;
    end;
  until ch='#';
  // выводим результат:
  // Он у нас будет в алфавитном порядке автоматически, так как английские буквы
  // во всех кодировках идут строго по алфавиту 1..26
  for i:=0 to 25 do if k[i]=max then write(Chr(i+Ord('A')),' ');
  writeln;
  writeln(max);
end.
```

Задача 10. На вход программе подаётся последовательность символов, заканчивающихся символом #. Другие символы # во входной последовательности отсутствуют.

Программа должна вывести на экран символы латинского алфавита, в порядке увеличения частоты встречаемости во входной последовательности.

Если буква во входной последовательности не встречается, то её выводить не нужно.

Если несколько букв встречаются одинаковое число раз, программа должна вывести их в алфавитном порядке.

Строчные и прописные буквы не различаются.

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна решать поставленную задачу.

Пример :

Входные данные:

Aced, ccedaa f#

Выходные данные:

FDEAC

Решение.

```
// PascalABC.NET 3.2
program ege27_10;
var
  ch: char;
  k: array[0..25] of integer; // 26 букв английского алфавита
  i,freq,max: integer;

begin
  for i:=0 to 25 do k[i]:=0; // не забудем обнулить все счётчики
  max:=0;
  repeat
    read(ch);
    ch:=UpCase(ch); // теперь символ в любом случае в верхнем регистре
    // Но там могут быть не только буквы, но и другие символы!
    i:=Ord(ch)-Ord('A');
    if (i>=0) and (i<=25) then // если это буква алфавита 0..25
    begin
      Inc(k[i]); // увеличиваем счётчик попаданий для данной буквы
      if k[i]>max then max:=k[i]; // заодно ищем максимальную частоту встречаемости
    end;
  until ch='#';
  // выводим результат:
  // Он у нас будет в алфавитном порядке автоматически, так как английские буквы
  // во всех кодировках идут строго по алфавиту 1..26
  for freq:=1 to max do for i:=0 to 25 do if k[i]=freq then write(Chr(i+Ord('A')));
end.
```

Задача 11. В уругвайской разведке работает N агентов (N<40 000). Каждый агент имеет свой собственный уникальный номер, совпадающий с порядковым номером ведомости на зарплату. Специальная пропускная система на входе фиксирует номер каждого агента, пришедшего на работу.

В середине дня начальник отдела кадров заподозрил, что, возможно, один из сотрудников на работу не пришёл. Начальник запросил у пропускной системы список пришедших сотрудников.

Вам нужно написать эффективную программу, которая определяет, все ли сотрудники уругвайской разведки пришли на работу, и вывести на экран список номеров всех пришедших агентов, отсортированный по порядку в зарплатной ведомости.

На вход программе подаётся:

- в первой строке: количество агентов в разведке N
- во второй строке: последовательность номеров сотрудников, заканчивающаяся нулём (этот ноль служит признаком окончания последовательности)

Программа должна вывести требуемую последовательность, если один из сотрудников не пришёл на работу, или сообщение "Все пришли", если прогульщиков нет.

Внимание! Здесь пример некорректной постановки задачи. Явно не сказано, но подразумевается (и это можно понять по намёкам): прогульщик если есть, то только один! Это типичная оторванная от жизни задача олимпиадного типа. В реальной жизни, естественно, не придти на работу может сколько угодно человек, и придётся по-честному сортировать список.

В данном случае подразумевается: массивов и списков не нужно, а нужно знать формулу: сумма всех целых чисел от 1 до N равна N*(N+1)/2.

Пример входных данных:

5

3 2 5 1 4 0

Выходные данные:

Все пришли

Другой пример входных данных:

5

3 5 1 4 0

Выходные данные:

1 3 4 5

Решение.

```
// PascalABC.NET 3.2
program ege27_11;
var
  x,N,sum: integer;
begin
  readln(N);
  sum:=N*(N+1) div 2; // сразу определим сумму всех номеров сотрудников
  repeat
    read(x);
    sum:=sum-x; // ноль для последнего не повлияет на результат
  until x=0;
  // так как прогульщик может быть только один (это кретинизм авторов задачи),
  // то значение s будет содержать его номер, или ноль, если все пришли
  if sum=0 then writeln('Все пришли')
  else for x:=1 to N do if x<>sum then write(x,' ');
end.
```

Задача 12. На вход программе подаются сведения о неуспеваемости ученика за весь период его обучения в школе, за каждый месяц в отдельности. В первой строке сообщается количество месяцев (N), которые проучился ученик. В каждой из N последующих строк: информация о количестве двоек, которые получил ученик за каждый месяц в течение всего периода обучения в формате: mmmm yyyy kkkk, где mmmm - название месяца (например, март или сентябрь), yyyy -год, kkk - число двоек.

Пример: **апрель 2006 12**

Порядок строк произвольный. Год - не раньше 1990 (может быть позже).

Возможно, некоторые месяцы ученик не учился и они отсутствуют в списке. Возможно, что некоторые годы ученик также не учился.

Напишите эффективную, в т.ч. по использованию памяти, программу, которая будет выводить на экран годы, в которые ученик учился лучше, чем в первый год своего обучения. Если таких нет, вывести об этом сообщение.

Считается, что ученик учится тем лучше, чем меньше у него двоек.

Пример входных данных:

6
январь 1995 12
февраль 1997 20
март 2001 5
апрель 2001 6
май 2005 10
июнь 2003 18

Программа должна вывести:

2001
2005

Внимание ! Пример некорректного условия в ЕГЭ! Подразумевается, хоть этого явно не сказано (а есть лишь намёк, по которому об этом можно догадаться), что годы сверху нужно ограничить текущим годом. Т.е. взять, к примеру, диапазон годов 1990..2018. Иначе задача резко усложняется, но это не уровень ЕГЭ.

```
// PascalABC.NET 3.2
program ege27_12;
var
  N,i,year,nd,min: integer;
  k: array[1990..2018] of integer; // количества двоек по годам
  ch: char;
begin
  for i:=1990 to 2018 do k[i]:=-1; // признак того, что год не заполнен
  min:=2018;
  readln(N);
  for i:=1 to N do
  begin
    // читаем символы месяца до пробела:
    repeat read(ch) until ch=' ';
    readln(year,nd); // читаем год и кол-во двоек
    if k[year]=-1 then k[year]:=nd else k[year]:=k[year]+nd;
    if year<min then min:=year; // заодно определим минимальный (первый) год
  end;
  nd:=0; // теперь это признак того,что печати не было
  // просматриваем все годы по порядку:
  for i:=1990 to 2018 do if (k[i]<>-1) and (k[i]<k[min]) then begin writeln(i); nd:=1;
end;
  if nd=0 then writeln('Таких годов не обнаружено');
end.
```

Задача 13. На вход программе подаются сведения о работе железнодорожной сортировочной станции, за каждый год в отдельности. В первой строке сообщается количество строк (N), которые будут поданы на вход программы.

В каждой из последующих N строк: информация о тоннаже и количестве железнодорожных вагонов, обработанных сортировочной станцией в формате:

mm.yyyy tttt kkk

где mm - номер месяца, yyyy - год, tttt - тоннаж, kkk - количество обработанных вагонов (число положительное, если вагоны прибывающие, отрицательное - если убывающие). Годы идут не раньше 1980 года.

Пример: 11.2001 45322 -8657

Необходимо вывести на экран номера лет, в которые наибольший тоннаж одного прибывшего вагона превышает наименьший тоннаж одного убывшего.

- Внимание к типам! Про округление ничего не сказано, значит, вычислять средний тоннаж вагона нужно вещественной арифметикой (real).
- И классика: подразумевается ограничение по годам 1980..(по текущий год).
- Наибольший/наименьший тоннаж за год - максимум/минимум показателя по месяцам этого года.
- Ответ написан под чтение из файла, на ЕГЭ этого не нужно - читать с клавиатуры!

Решение.

```
// PascalABC.NET 3.2
program ege27_13;
var
  f: text;
  i,N,year,tonn,number: integer;
  mid: real;
// наиб. тоннаж для 1 прибывшего вагона и наименьш. - для убывшего за каждый год:
  tonn_in_max,tonn_out_min: array[1980..2018] of real;
  // real, чтобы не потерять точность при усреднении тоннажа
  ch: char;
 begin
   Reset(f,'data1.txt');
   readln(f,N); // читаем кол-во строк
   for i:=1980 to 2018 do begin tonn_in_max[i]:=0; tonn_out_min[i]:=0; end;
   for i:=1 to N do
   begin
     read(f,ch,ch,ch); // пропускаем первые три символа - месяц и точку
     readln(f,year,tonn,number);
     mid:= Abs(tonn/number); // средний тоннаж 1 вагона точно, без округления
     //-------------------------------------------------------------------------
     // начало длинной вложенной конструкции if:
     if number>0 then // для прибывающих вагонов:
     begin // здесь begin/end обязательны, чтобы else относился к нужному if
       if mid>tonn_in_max[year] then tonn_in_max[year]:=mid; // ищем макс. для года
     end
     else // для убывающих вагонов
     // внимание! начальное значение не задано, т.к. непонятно чему задавать
     // поэтому для определения факта первого значения используем значение 0!
     if tonn_out_min[year]=0 then tonn_out_min[year]:=mid // для первого раза!
     else // а для последующих будем проверять:
     if mid<tonn_out_min[year] then tonn_out_min[year]:=mid;
     // конец длинной вложенной конструкции if
     //-------------------------------------------------------------------------
   end;
   // выводим результат:
   for i:=1980 to 2018 do if tonn_in_max[i]>tonn_out_min[i] then writeln(i);
   Close(f);
 end.
```

Ответ к тесту data1.txt:

2005

2008

Задача 14. На вход программы подаётся: в первой строке - количество входных чисел N.

В последующих N строках - последовательность из N целых чисел. Известно, что каждое число положительное и не превышает 10^9.

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна вывести на экран максимальное произведение двух различных элементов последовательности, которое кратно 6.

Под "различными" следует понимать не различные значения, а различные номера в последовательности. То есть, результат может быть квадратом некоторого числа, если оно в последовательности встречается не менее двух раз (и при этом максимально).

Если такой пары элементов нет, программа должна вывести ноль.

Пример входных данных:

4
9
10
29
3

Программа должна вывести:

90

Решение. Это иезуитская задача олимпиадного типа, очень коварная. Сначала нужно немного порассуждать.

Ясно, что запоминать все числа в массиве нельзя. Это назовут "неэффективностью по памяти" и снизят балл. Хотя если времени вообще нет, можно тупо запомнить все числа в массив, размер которого указать от балды, и тупо перебрать все различные номера (i,j) на предмет указанного максимума. Это просто и даст какой-то балл, так что как аварийный вариант нужно иметь ввиду.

В таких задачах всегда подразумевается фильтрация входного потока и конструирование ответа в конце. Поэтому подумаем, как бы мы действовали, если бы не было условия кратности 6: мы бы нашли максимум и второй максимум (при допущении их совпадения по значениям) и выдали бы ответ как их произведение. Очевидно, что произведение двух максимальных в последовательности числел и будет решением задачи.

Что меняет добавление условия кратности произведения 6?

Последовательность произвольная, поэтому возможны любые варианты. Значит, нужно их просто рассмотреть все. Начнём с конца. Наше произведение делится на 6. Оно должно делиться на 6, поэтому мы так и говорим - оно делится. Как это может быть обеспечено? Здесь 2 варианта, потому что 6=2*3 - не простое число:

А) Один из сомножителей делится на 6.

Б) Один из сомножителей делится на 2, а второй при этом - делится на 3. Тогда в их произведении образуется 2*3=6.

Рассмотрим вариант А. Поскольку нас интересует максимум произведения, в первую очередь нас будет интересовать 1-й максимум последовательности. Он у нас по-любому должен войти в произведение, при этом возможно 2 варианта:

А1) 1-й максимум делится на 6. Это самый лучший случай. Тогда мы просто выбираем его и 2-й максимум, свойства которого нам уже неважны. Ответ будет гарантировано максимальным и кратным 6.

А2) Иначе (1-й максимум НЕ делится на 6). Вторым сомножителем придётся взять (максимум, кратный 6). Иначе 6-ке взятся неоткуда (в пределах варианта А).

Рассмотрим вариант Б. Ни один из сомножителей не делится на 6. Это означает, что (максимум, кратный 2) и (максимум, кратный 3) - различные элементы последовательности (иначе бы они совпали с максимумом, кратным 6). В этом случае ответ будет равен произведению (максимум, кратный 2) * (максимум, кратный 3), которое гарантировано делится на 6.

Сразу сказать, ответ по какому варианту будет больше, нельзя. Поэтому придётся в конце сравнить

эти варианты.

Итак, при фильтрации потока чисел нам нужно будет найти величины:

- максимум 1-й
- максимум 2-й (даже совпадающий с 1-м)
- (максимум, кратный 6)
- (максимум, кратный 2)
- (максимум, кратный 3)

Зная эти величины, мы всегда можем сконструировать ответ.

```
// PascalABC.NET 3.2
program ege27_14;
var
  i,x,N,res: integer;
  max,max2nd: integer; // абсолютные чемпионы 1 и 2 место
  max2,max3,max6: integer; // максимумы, кратные 2,3,6
begin
  // обнулим максимумы, т.к. числа положительные, т.е. нулей быть не может
  // кроме того, нулевое значение даст ноль в ответе при отсутствии результата
  max:=0;
  max2nd:=0;
  max2:=0; // и не кратный 6!
  max3:=0; // и не кратный 6!
  max6:=0;
  readln(N);
  for i:=1 to N do
  begin
    readln(x);
    // ищем абсолютные максимумы 1 и 2-й:
    if x>max then begin max2nd:=max; max:=x; end
    else if x>max2nd then max2nd:=x;
    // ищем максимумы, кратные 6,3,2:
    if x mod 6=0 then begin if x>max6 then max6:=x; end // begin/end обязательны!
    else if x mod 3=0 then begin if x>max3 then max3:=x; end
         else if x mod 2=0 then if x>max2 then max2:=x;
  end;
  // конструируем ответ. Сначала рассмотрим случай А: искомое произведение
  // содержит один из сомножителей, кратный 6. При этом возможно 2 случая:
  // 1) max не кратен 6. Тогда его нужно умножить на max6
  // 2) кратен. Тогда умножаем его на max2nd не глядя.
  if max=max6 then res:=max*max2nd // простая проверка, кратен ли max 6?
  else             res:=max*max6;  // не кратен
  // Теперь учтём случай Б. Тут всё проще - искомое произведение max2*max3.
  // Поэтому сразу сравним оба случая:
  if max2*max3>res then writeln(max2*max3)
  else writeln(res);
  // Ноль появится автоматически, если максимумов не будет обнаружено
end.
```

Задача 15. На вход программы подаётся: в первой строке - количество входных чисел N.

В последующих N строках - последовательность из N целых чисел. Известно, что каждое число положительное и не превышает 10^9.

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая должна вывести на экран максимальное произведение двух различных элементов последовательности, которое **не** кратно 15.

Под "различными" следует понимать не различные значения, а различные номера в последовательности. То есть, результат может быть квадратом некоторого числа, если оно в последовательности встречается не менее двух раз (и при этом максимально).

Если такой пары элементов нет, программа должна вывести ноль.

Пример входных данных:

4
90
10
29
3

Программа должна вывести:

```
290
```

Решение. Ещё более иезуитская задача. Сразу обращаем внимание на то, что 15=3*5. Поэтому всё внимание должно быть на кратность 3 и 5, а не 15 (если число не кратно 3, то оно подавно не кратно 15, то же самое с 5). Однако может получиться, что один из множителей будет кратен 3, а второй будет кратен 5, в этом случае произведение будет кратно 15.

Поэтому здесь придётся тупо рассматривать все варианты максимумов:

1) Максимумы 1-2, которые не кратны ни 3, ни 5
2) Максимумы 1-2, которые кратны только 3 (но не 5)
3) Максимумы 1-2, которые кратны только 5 (но не 3)

Из этих значений придётся рассмотреть 5 комбинаций, выбрав из них максимальный:

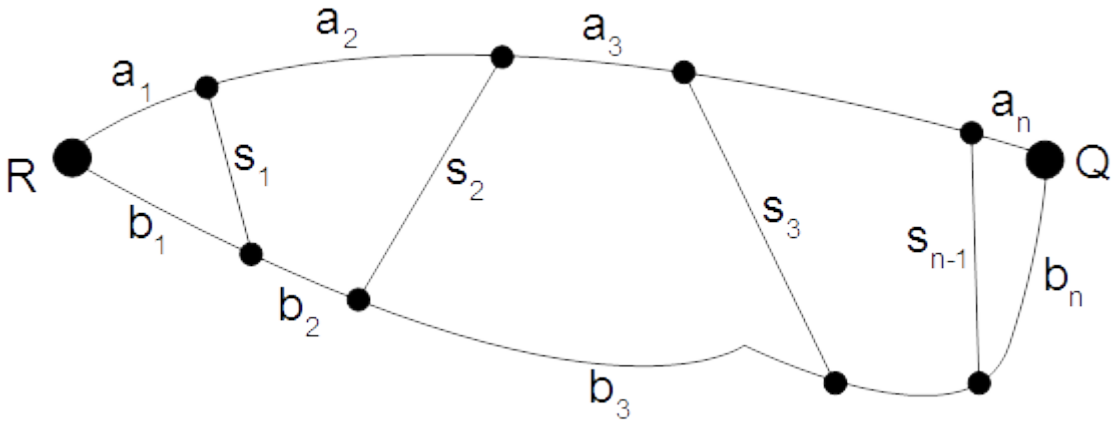
1. Перемножаем максимумы 1-2, не кратные ни 3, ни 5
2. (максимум1, не кратный ни 3 ни 5)*(максимум1, кратный только 3)
3. (максимум1, не кратный ни 3 ни 5)*(максимум1, кратный только 5)
4. (максимум1, кратный только 3)*(максимум2, кратный только 3)
5. (максимум1, кратный только 5)*(максимум2, кратный только 5)

```
// PascalABC.NET 3.2
program ege27_15;
var
  i,x,N,res: integer;
  max,max_2,max3,max3_2,max5,max5_2: integer;
begin
  // группа чистых чемпионов:
  max:=0; // максимум, не кратный ни 3 ни 5
  max_2:=0;  // второй максимум, не кратный ни 3 ни 5
  // группа чемпионов похуже, в которых допускаются 3-ки, но нет 5-ок:
  max3:=0; // максимум, кратный 3, но не кратный 5
  max3_2:=0; // второй максимум, кратный 3, но не кратный 5
  // группа чемпионов похуже, в которых допускаются 5-ки, но нет 3-ек:
  max5:=0; // максимум, кратный 5, но не кратный 3
  max5_2:=0; // второй максимум, кратный 5, но не кратный 3
  readln(N);
  for i:=1 to N do
  begin
    readln(x);
    // ищем чистых чемпионов:
    if (x mod 3<>0) and (x mod 5<>0) then
      if x>max then begin max_2:=max; max:=x; end
      else if x>max_2 then max_2:=x;
    // ищем чемпионов с 3-ми, но без 5-ок:
    if (x mod 3=0) and (x mod 5<>0) then
      if x>max3 then begin max3_2:=max3; max3:=x; end
      else if x>max3_2 then max3_2:=x;
    // ищем чемпионов с 5-ми, но без 3-ек:
    if (x mod 3<>0) and (x mod 5=0) then
      if x>max5 then begin max5_2:=max5; max5:=x; end
      else if x>max5_2 then max5_2:=x;
  end;
  // последовательность обработали, перебираем 5 возможный вариантов ответа:
  res:=max*max_2; // допустим, что это произведение двух чистых чемпионов
  if max*max5>res then res:=max*max5;
  if max*max3>res then res:=max*max3;
  if max3*max3_2>res then res:=max3*max3_2;
  if max5*max5_2>res then res:=max5*max5_2;
  writeln(res); // ноль получился сам, если чего не будет найдено
end.
```

Задача 16. Из населённого пункта R в населённый пункт Q ведут две дороги. Назовём их A и B.

Дороги идут невдалеке друг от друга, не пересекаются. Периодически между ними встречаются соединительные дороги (связки), по которым можно переехать с дороги A на дорогу B и обратно.

Необходимо найти кратчайшее расстояние между населёнными пунктами R и Q при условии, что ехать можно по любой дороге - A или B, и любое количество раз переезжать (если это нужно) по дорогам-связкам с одной дороги на другую.



На вход программе подаётся: в первой строке - число дорог-связок N.

В каждой из последующих N строк - три целых неотрицательных числа:

1) расстояние от предъидущей "развилки" по дороге A
2) расстояние от предъидущей развилки по дороге B
3) длина дороги-связки (в последней строке это будет 0).

Пример входных данных:

4
3 7 3
12 2 4
3 10 5
7 6 0

Программа должна вывести:

22

```
// PascalABC.NET 3.2
program ege27_16;
var
  N,i,a,b,a1,b1,s1,abest: integer;
begin
  a:=0;  // кратчайшее расстояние от начала до текущего узла дороги A
  b:=0;  // кратчайшее расстояние от начала до текущего узла дороги B
  readln(N);
  for i:=1 to N do
  begin
    readln(a1,b1,s1);
    // считали параметры текущего сегмента. Наша задача - определить
    // значения (a,b) - кратчайшие расстояния до узлов этого нового сегмента
    // считаем кратчайший путь до узла на дороге A:
    if a+a1 < b+b1+s1 then abest:=a+a1 // едем по дороге A в этот узел дороги A
    else abest:=b+b1+s1;               // едем по дороге B в этот узел дороги A
    // значение a пока не трогаем, чтобы не нарушить расчёт для второго узла
    // теперь считаем кратчайший путь до узла на дороге B:
    if b+b1 < a+a1+s1 then b:=b+b1 // едем по дороге B в этот узел дороги B
    else b:=a+a1+s1;               // едем по дороге A в этот узел дороги B
    a:=abest; // меняем a на наилучший вариант
    // Теперь у нас есть значения (a,b) для этого сегмента
  end;
  writeln(a); // заметим,что при s1=0 a и b должны совпасть.
end.
```

Задача 17(2). На вход программе подаётся: в первой строке - число N ($5< N < 10^9$).

В каждой из следующих N строк - по одному элементу последовательности - натуральные числа, не превышающие 10^9.

Напишите программу, эффективную в т.ч. по используемой памяти (укажите используемую версию языка программирования), которая выводит на экран максимальную сумму двух элементов этой последовательности, *номера* которых различаются на *5*.

Пример входных данных:

8
3
4
5
7
8
3
2
9

Программа должна вывести:

14

```
// PascalABC.NET 3.2
program ege27_17_2;
var
  N,i,j,elem,max: integer;
  mem: array[1..5] of integer; // очередь из последних 5 значений элементов
  // 1 - самый первый пришедший, 5 - последний пришедший
begin
  max:=0; // макс.сумма. Поскольку числа натуральные, 0 заведомо меньше любого
  readln(N);
  // сначала заполняем очередь. Пока она не заполнена, работать не с чем:
  for i:=1 to 5 do readln(mem[i]); // гарантировано,что там есть более 5 элементов
  // работаем с оставшимися элементами:
  for i:=6 to N do
  begin
    readln(elem);
    if elem+mem[1]>max then max:=elem+mem[1];
    for j:=1 to 4 do mem[j]:=mem[j+1]; // сдвигаем очередь
    mem[5]:=elem; // и добавляем новый элемент в её конец
  end;
  writeln(max);
end.
```

Задача 17. На вход программе подаётся: в первой строке - число N ($5< N < 10^9$).

В каждой из следующих N строк - по одному элементу последовательности - натуральные числа, не превышающие 10^9.

Напишите программу, эффективную в т.ч. по используемой памяти (укажите используемую версию языка программирования), которая выводит на экран максимальную сумму двух элементов этой последовательности, *номера* которых различаются *не меньше, чем на 5*.

Пример входных данных:

8
3
4
5
7
8
3
2
9

Программа должна вывести:

14

Решение. Задача сложнее, потому что зона поиска становится постоянно расширяющейся. Нарисуйте последовательность и выделите зону поиска - на что нужно домножать текущий элемент, чтобы получилось максимальное произведение? Ясно, что на максимум из зоны поиска.

```
// PascalABC.NET 3.2
program ege27_17;
var
  N,i,j,elem: integer;
  max: integer; // текущий максимум из зоны поиска: 1..(i-5)
  maxSum: integer; // максимальное значение суммы max и текущего элемента
  mem: array[1..5] of integer; // очередь из последних 5 значений элементов
  // 1 - самый первый пришедший, 5 - последний пришедший
  // 1 элемент постоянно обновляет max из зоны поиска.
  // Это максимум из всех возможных элементов, отстоящих от текущего на
  // расстояние 5 и более. Именно на него и нужно умножать текущий
begin
  readln(N);
  // сначала заполняем очередь:
  for i:=1 to 5 do readln(mem[i]); // гарантировано,что там есть более 5 элементов
  // работаем с оставшимися элементами:
  for i:=6 to N do
  begin
    readln(elem);
    if i=6 then begin max:=mem[1]; maxSum:=max+elem; end // задаём начальные знач.
    else begin // если не начальные, то обновляем максимумы:
      if mem[1]>max then max:=mem[1]; // это максимум из всей доступной зоны поиска
      if max+elem>maxSum then maxSum:=max+elem;
    end;
    for j:=1 to 4 do mem[j]:=mem[j+1]; // сдвигаем очередь
    mem[5]:=elem; // и добавляем новый элемент в её конец
  end;
  writeln(maxSum);
end.
```

Задача 18. Вам предлагается два задания, связанные с этой задачей: задание А и задание Б. Вы можете решать оба задания или одно из них по выбору.

Итоговая оценка выставляется как максимум из оценок за А и Б. Если решения одного из заданий нет, то считается, что оценка за него 0.

Задание Б является усложнённым вариантом задания А, оно содержит дополнительные требования к программе.

Напишите программу, которая должна вывести на экран минимальное чётное произведение двух элементов последовательности, номера которых различаются не менее, чем на 6.

Если такой пары элементов нет, программа должна вывести ноль.

Задание А. Напишите программу для решения поставленной задачи, в которой входные данные будут запоминаться в массиве, после чего будут проверены все пары элементов. Перед программой укажите вариант задания - А и версию языка программирования.

Максимальная оценка за задание А - 2 балла.

Задание Б. Напишите программу для решения поставленной задачи, которая будет эффективна как по времени, так и по памяти (или хотя бы по одной из этих характеристик).

Программа считается эффективной по памяти, если размер памяти, использованной в программе для хранения данных, не зависит от числа N и не превышает 1 Кбайта.

Перед программой укажите версию языка программирования и кратко опишите использованный алгоритм. Обязательно укажите, что программа является решением задания Б.

Максимальная оценка за правильную программу, эффективную по времени и по памяти - 4 балла.

Максимальная оценка за правильную программу, эффективную по времени, но неэффективную по памяти - 3 балла.

Входные данные представлены следующим образом. В первой строке задаётся число N - общее количество элементов последовательности ($N \le 10\,000$).

В каждой из следующих N строк задаётся одно положительное число - очередной элемент последовательности.

Известно, что каждое число положительное и не превышает 1000.

Пример входных данных:

8
1
4
5
7
8
3
2
9

Программа должна вывести:

2

Комментарий: настоятельно рекомендуется в условиях дефицита времени решать задачу А. Там элементарное, шаблонное решение по перебору, но оно позволяет получить 2 балла, что неплохо.

В этом задании эффективность не оценивается, поэтому не нужно мудрить с оптимизацией перебора! Да, там можно сократить область перебора и чуть оптимизировать вычисления. Но это усложнит программу, откроет двери массе ошибок, которые легко допустить в стрессовом состоянии. Поэтому - задание А пишите просто, тупо, главное - правильный результат!

```
// Решение задания А
// PascalABC.NET 3.2
program ege27_18a;
var
  N,i,j,minp: integer;
  mem: array[1..10000] of integer;
begin
  readln(N);
  for i:=1 to N do readln(mem[i]);
  // перебираем все возможные пары элементов:
  minp:=1000*1000+1; // не забудем!! Максимум нам известен = 1000
  for i:=1 to N do
    for j:=1 to N do
      if (abs(i-j)>=6) and (mem[i]*mem[j] mod 2=0) and (mem[i]*mem[j]<minp) then
      minp:=mem[i]*mem[j];
  if minp=1000*1000+1 then writeln(0)
  else writeln(minp);
end.
```

```
// Решение задания Б
// PascalABC.NET 3.2
program ege27_18b;
const START: integer = 1000*1000+1; // начальное значение искомого минимума
var
  N,i,j,elem,minp,min,min2: integer;
  mem: array[1..6] of integer;
begin
  readln(N);
  for i:=1 to 6 do readln(mem[i]); // заполняем очередь
  min:=1001; // минимум из зоны поиска без учёта чётности
  min2:=1001; // чётный минимум из зоны поиска
  minp:=START; // искомый минимум
  for i:=7 to N do
  begin
    // Зона поиска сейчас - от 1 элемента до mem[1]
    // Поэтому учтём появление в ней нового элемента:
   if mem[1]<min then min:=mem[1]; // это минимум без учёта чётности
   if mem[1] mod 2=0 then if mem[1]<min2 then min2:=mem[1];
   // теперь зона поиска актуальна, и можно читать след.элемент:
   readln(elem);
   // Действовать будем в зависимости от его чётности.
   // Если он чётный, то неважно, чётный или нет второй элемент
   // Если же он нечётный, то его можно домножать только на чётный минимум:
   if elem mod 2=0 then
     begin if elem*min<minp then minp:=elem*min end
   else    if elem*min2<minp then minp:=elem*min2;
   // сдвигаем очередь:
   for j:=1 to 5 do mem[j]:=mem[j+1];
   // и дописываем новый элемент в её конец:
   mem[6]:=elem;
  end;
  if minp=START then writeln(0) else writeln(minp);
end.
```

Задание 19. На вход программе подаётся: в первой строке - число N ($1 < N < 10^9$).

В каждой из следующих N строк - по одному элементу последовательности - целые числа, не превышающие по модулю 10 000.

Напишите программу, эффективную в т.ч. по используемой памяти (укажите используемую версию языка программирования), которая выводит на экран минимальное произведение двух элементов этой последовательности.

Пример входных данных:

8
3
4
5
-7
-8
3
2
9

Программа должна вывести:

-72

Решение. В задаче подвох, связанный с наличием отрицательных чисел. Можно делать разными способами, но в данном случае лучше максимально просто:

- находим 1 и 2 максимумы (max и max2)
- находим 1 и 2 минимумы (min и min)

Потом проверяем возможные комбинации, на самом деле их возможно всего три:

1) либо ответ = max*max2 (положительных чисел нет)
2) либо ответ = min*min2 (отрицательных чисел нет)
3) либо ответ = min*max (есть и те и другие)

Наличие нулей на решение не влияет, что нужно проверить расуждениями.

```
// PascalABC.NET 3.2
program ege27_19;
var
  N,i,elem,res: integer;
  max,max2,min,min2: integer;
begin
  readln(N);
  max:=-10001;
  min:=10001;
  for i:=1 to N do
  begin
    readln(elem);
    // ищем 1 и 2 максимумы:
    if elem>max then begin max2:=max; max:=elem; end
    else if elem>max2 then max2:=elem;
    // ищем 1 и 2 минимумы:
    if elem<min then begin min2:=min; min:=elem; end
    else if elem<min2 then min2:=elem;
  end;
  // формируем ответ, рассмативая 3 возможных варианта:
  res:=max*max2; // если положительных чисел не было, то так
  if min*min2<res then res:=min*min2; // если отрицательных чисел не было
  if min*max<res then res:=min*max; // если были и те и другие
  writeln(res);
end.
```

Задача 20. На вход программы подаётся последовательность символов, заканчивающаяся символом #. Других символов # во входной последовательности нет.

Будем называть словом любую последовательность подряд идущих символов, отделённую пробелами. Также слово может начинаться с начала строки.

Известно, что во входной последовательности не более 10^9 символов, при этом слова имеют длину не более 100 символов.

Напишите эффективную, в т.ч. и по используемой памяти, программу (укажите версию языка программирования), которая найдёт и выведет на экран слово максимальной длины. Если во входной строке несколько слов максимальной длины, необходимо вывести на экран первое такое слово.

Пример входных данных:

Если видишь в небе люк, не волнуйся, это глюк. Если люк ты видишь в крыше, значит, лезь скорей повыше.#

Программа должна вывести:

волнуйся,

Еще пример входных данных:

Day, mice. "Year" - a mistake#

Программа должна вывести:

mistake

Комментарий:

1) обратите внимание на то, что признаком окончания слова является не только пробел, но и знак #. Здесь - тонкое место, где нужно не пропустить проверку последнего слова, которое заканчивается решёткой, как во втором примере. Алгоритм нужно построить так, чтобы эта проверка выполнялась в обоих случаях.
2) если указано, что длина строки не превышает 100, то нужно использовать переменные типа string[100] для экономии памяти и времени. Если сделать string, то расход памяти будет выше, и может быть больше времени выполнения на динамическое выделение памяти.
3) Для строк работает операция прибавления, которая приклеивает к строке символ справа. То есть для переменных ch: char и str: string[100] вполне можно написать str:=str+ch; - тип char Нормально прибавится к string, потому что string на самом деле - это массив из char. Есть еще сокращённая форма этой операции: str+=ch;

Решение.

```
// PascalABC.NET 3.2
program ege27_20;
var
  ch: char;  // текущий входной символ
  w,maxw: string[100]; // текущее слово и слово максимальной длины
begin
  w:=''; maxw:='';
  repeat
    read(ch);
    if (ch=' ') or (ch='#') then // текущее слово закончилось и лежит в w:
    begin
     if Length(w)>Length(maxw) then maxw:=w; // проверим на максимум
     w:=''; // не забудем обнулить текущее слово
    end
    else  w:=w+ch;  // прибавляем текущий символ к строке справа
  until ch='#';
  writeln(maxw);
end.
```