**Демонстрационный вариант ЕГЭ 2018 – задание №27** — На вход программы поступает последовательность из N целых положительных чисел, все числа в последовательности различны.

Рассматриваются все пары различных элементов последовательности (элементы пары не обязаны стоять в последовательности рядом, порядок элементов в паре не важен). Необходимо определить **количество пар**, для которых **произведение элементов делится на 26**.

**Описание входных и выходных данных**

В первой строке входных данных задаётся количество чисел N (1 ≤ N ≤ 1000). В каждой из последующих N строк записано одно целое положительное число, не превышающее 10 000.

В качестве результата программа должна напечатать одно число: количество пар, в которых произведение элементов кратно 26.

*Пример входных данных:*

4
2
6
13
39

*Пример выходных данных для приведённого выше примера входных данных:*

4

*Пояснение*. Из четырёх заданных чисел можно составить 6 попарных произведений: 2·6, 2·13, 2·39, 6·13, 6·39, 13·39 (результаты: 12, 26, 78, 78, 234, 507). Из них на 26 делятся 4 произведения (2·13=26; 2·39=78; 6·13=78; 6·39=234).

Требуется написать эффективную по времени и по памяти программу для решения описанной задачи.

Программа считается эффективной по времени, если при увеличении количества исходных чисел N в k раз время работы программы увеличивается не более чем в k раз.

Программа считается эффективной по памяти, если память, необходимая для хранения всех переменных программы, не превышает 1 Кбайт и не увеличивается с ростом N.

Максимальная оценка за правильную (не содержащую синтаксических ошибок и дающую правильный ответ при любых допустимых входных данных) программу, эффективную по времени и по памяти, – 4 балла.

Максимальная оценка за правильную программу, эффективную только по времени – 3 балла.

Максимальная оценка за правильную программу, не удовлетворяющую требованиям эффективности, – 2 балла.

Вы можете сдать **одну** программу или **две** программы решения задачи (например, одна из программ может быть менее эффективна). Если Вы сдадите две программы, то каждая из них будет оцениваться независимо от другой, итоговой станет **бóльшая** из двух оценок.

Перед текстом программы обязательно кратко опишите алгоритм решения.

Укажите использованный язык программирования и его версию.

**Решение:**

Произведение двух чисел делится на 26, если выполнено одно из следующих условий (условия не могут выполняться одновременно).

**А. Оба сомножителя делятся на 26.**

**Б. Один из сомножителей делится на 26, а другой не делится.**

**В. Ни один из сомножителей не делится на 26, но один сомножитель делится на 2, а**

**другой – на 13.**

Примечание для проверяющего. Условие делимости произведения на 26 можно сформулировать проще, например, так: (один из сомножителей делится на 26) ИЛИ (один сомножитель делится на 2, а другой – на 13).

Но в этом случае пара сомножителей может удовлетворять обоим условиям, что затруднит подсчёт количества пар.

При вводе чисел можно определять, делится ли каждое из них на 26, 2 и 13, и подсчитывать следующие значения:

1) n26 – количество чисел, кратных 26;

2) n13 – количество чисел, кратных 13, но не кратных 26;

3) n2 – количество чисел, кратных 2, но не кратных 26.

Примечание для проверяющего. Сами числа при этом можно не хранить.

Каждое число учитывается не более чем в одном из счётчиков.

Количество пар, удовлетворяющих условию А, можно вычислить по формуле **n26·(n26 – 1)/2.**

Количество пар, удовлетворяющих условию Б, можно вычислить по формуле **n26·(N – n26).**

Количество пар, удовлетворяющих условию В, можно вычислить по формуле **n2·n13.**

Поэтому искомое количество пар вычисляется по формуле

n26·(n26 – 1)/2 + n26·(N – n26) + n2·n13.

Ниже приведена реализующая описанный алгоритм программа на языке Паскаль (использована версия PascalABC)

```
var
N: integer; {количество чисел}
a: integer; {очередное число}
n26, n13, n2: integer;
k26: integer; {количество требуемых пар}
i: integer;
begin
readln(N);
n26:=0; n13:=0; n2:=0;
for i:=1 to N do begin
        readln(a);
if a mod 26 = 0 then
        n26 := n26+1
else if a mod 13 = 0 then
        n13 := n13 + 1
else if a mod 2 = 0 then
        n2 := n2 + 1;
end;
k26 := n26*(n26-1) div 2 + n26*(N-n26) + n2*n13;
writeln(k26)
end.
```

**Программа на языке C++. (4б)**

```
#include <iostream>
using namespace std;
int main() {
int N;     //количество чисел
int a;  //очередное число
int n26, n13, n2 ;
int k26;  //количество требуемых пар
int i;
```

```
cin >> N;
n26=0; n13=0; n2=0;
for (i = 0; i<N; i++){
        cin >>  a;
        if (a % 26 == 0)
                n26 = n26+1;
        else if (a % 13 == 0)
                n13 = n13 + 1;
        else if (a % 2 == 0)
                n2 = n2 + 1;
}
k26 = n26*(n26-1)/2 + n26*(N-n26) + n2*n13;
cout << k26;
return 0;
}
```

Комментарии для проверяющего
1. При таком решении каждое прочитанное число обрабатывается (делаются проверки делимости, изменяются счётчики) и после этого не хранится.
Таким образом, используемая память не зависит от длины последовательности. Время обработки очередного числа фиксировано, т.е. не зависит от длины последовательности. Время заключительных
вычислений по приведённой в решении формуле также не зависит от длины последовательности. Поэтому при увеличении длины последовательности в k раз время работы программы увеличивается не более чем в k раз. Таким образом, приведённая выше программа эффективна как по времени, так и по используемой памяти. Это решение оценивается 4 баллами.
2. Общая идея решения, эффективного по времени, состоит в следующем.
Просматриваем по очереди все элементы последовательности и накапливаем значения вспомогательных величин (в приведённом решении это счётчики n2, n13, n26). После того как вся последовательность обработана и подсчитаны окончательные значения вспомогательных величин, по этим значениям подсчитывается искомое количество пар.
При этом можно использовать и другие вспомогательные величины.
Например, можно вместо n2 и n13 использовать величины p2 и p13 – количества чисел, которые делятся соответственно на 2 и на 13. Так как n2 = p2 – n26 и n13 = p13 – n26, то итоговая формула примет вид:
n26·(n26 – 1)/2 + n26·(N – n26) + (p2 – n26)·(p13 – n26).

Ещё один возможный вариант (есть и другие!) – подсчёт количества чисел, которые не делятся на 26, – можно вести по формуле n2+n13+nx, где nx – количество чисел, которые не делятся ни на 2, ни на 13. Значение nx можно вычислить с помощью отдельного счётчика. Такая программа на языке Бейсик приведена ниже.
Все подобные программы оцениваются в 4 балла.
При любом наборе вспомогательных величин возможны различные способы записи итоговой формулы. Можно, например, раскрывать скобки и приводить подобные члены или, наоборот, выносить за скобки общие
множители; можно вводить дополнительные переменные для отдельных слагаемых, а затем вычислять их сумму. Допустим любой способ записи вычислений, эквивалентный правильной формуле, выбранный способ записи не влияет на оценку.
3. Возможно решение, основанное на описанных идеях, однако предварительно сохраняющее элементы последовательности в массив. Такое решение (если в нём нет ошибок) эффективно по времени, но неэффективно по памяти. Оно оценивается в 3 балла.

4. Решение, не эффективное ни по времени, ни по памяти, запоминает входную последовательность в массиве, после чего явно перебирает все возможные пары. Такое решение оценивается в 2 балла (см. критерии)

Пример 2. Программа на языке Бейсик. Программа эффективна по времени и по памяти, но использует формулы, отличные от формул программы из примера 1

```
N26 = 0
N2 = 0
N13 = 0
NX = 0
INPUT N
FOR I = 1 TO N
        INPUT A
IF A MOD 26 = 0 THEN
        N26 = N26 + 1
  ELSE
        IF A MOD 13 = 0 THEN
                N13 = N13 + 1
        ELSE
          IF A MOD 2 = 0 THEN
                N2 = N2 + 1
          ELSE NX = NX + 1
          END IF
        END IF
  END IF
NEXT I
K26 = N26*(N26-1)\2 + N26*(N2+N13+NX) + N2*N13
PRINT K26
```

**Пример 3. Программа на языке C++. (2б)**

```
#include <iostream>
using namespace std;

int main() {
        int a[1000];
        int i, j, N, k=0;
        cin>>N;
        for(i=0;i<N;i++)
                cin>>a[i];
        for(i=0;i<N-1;i++)
                for(j=i+1;j<N;j++)
                        if(a[i]*a[j]%26==0)
                                k++;
        cout<<k;
        return 0;
}
```

**Пример 3. Программа на языке Python. (2б)**

```
a = []
n = int(input())
for i in range(0, n):
    a.append(int(input()))
d = 1
count = 0
```

```
for i in range(0, n - d):
    for j in range(i + d, n):
        if a[i] * a[j] % 26 == 0:
            count += 1
print(count)
```

---

## Демонстрационный вариант ЕГЭ 2017 г. – задание №27

Вам предлагается два задания с похожими условиями: задание А и задание Б. Вы можете решать оба задания или одно из них по своему выбору. Задание Б более сложное, его решение оценивается выше.
Итоговая оценка выставляется как **максимальная** из оценок за задания А и Б.

**Задание А.** Имеется набор данных, состоящий из **6 пар положительных** целых чисел.
Необходимо **выбрать из каждой пары ровно одно число** так, чтобы **сумма всех выбранных чисел не делилась на 3** и при этом была **максимально возможной**. Если получить требуемую сумму невозможно, в качестве ответа нужно выдать .
Напишите программу для решения этой задачи. В этом варианте задания оценивается только правильность программы, время работы и размер использованной памяти не имеют значения.
Максимальная оценка за правильную программу – 2 балла.

**Задание Б.** Имеется набор данных, состоящий из пар положительных целых чисел. Необходимо выбрать **из каждой пары ровно одно число** так, чтобы **сумма всех выбранных чисел не делилась на 3** и при этом была **максимально возможной**. Если получить требуемую сумму невозможно, в качестве ответа нужно выдать .
Напишите программу для решения этой задачи.
Постарайтесь сделать программу эффективной по времени и используемой памяти (или хотя бы по одной из этих характеристик).
Программа считается эффективной по времени, если время работы программы пропорционально количеству пар чисел N, т.е. при увеличении N в k раз время работы программы должно увеличиваться не более чем в k раз.
Программа считается эффективной по памяти, если размер памяти, использованной в программе для хранения данных, не зависит от числа N и не превышает 1 килобайта.
Максимальная оценка за правильную программу, эффективную по времени и памяти, – 4 балла.
Максимальная оценка за правильную программу, эффективную по времени, но неэффективную по памяти, – 3 балла.

Как в варианте А, так и в варианте Б программа должна напечатать одно число – максимально возможную сумму, соответствующую условиям задачи (или 0, если такую сумму получить нельзя).
НАПОМИНАЕМ! Не забудьте указать, к какому заданию относится каждая из представленных Вами программ.

Перед текстом программы кратко опишите Ваш алгоритм решения, укажите использованный язык программирования и его версию (например, Free Pascal 2.6.4).

*Входные данные*

Для варианта А на вход программе подаётся шесть строк, каждая из которых содержит два натуральных числа, не превышающих 10 000.
*Пример входных данных для варианта А:*
1 3
5 12
6 9
5 4
3 3
1 1

Для варианта Б на вход программе в первой строке подаётся количество пар N (1 ≤ N ≤ 100 000). Каждая из следующих N строк содержит два натуральных числа, не превышающих 10 000.
*Пример входных данных для варианта Б:*
6
1 3
5 12
6 9
5 4
3 3
1 1
*Пример выходных данных для приведённых выше примеров входных данных:*
32

**Решение:**

***Задание Б.*** Сначала рассмотрим решение для более общего задания (вариант Б).
**Решение 1.**
Чтобы получить максимально возможную сумму, **будем брать из каждой пары самое большое число**. Если полученная при этом сумма будет делиться на 3, её необходимо уменьшить. Для этого достаточно в одной из пар, где числа имеют разные остатки при делении на 3, заменить ранее выбранное число на другое число из той же пары. При этом **разница между числами в паре должна быть минимально** возможной. **Если во всех парах оба числа имеют одинаковый остаток при делении на 3, получить нужную сумму невозможно.**
**Замечание для эксперта.** От ученика не требуется доказывать правильность предложенного алгоритма. Для удобства экспертов докажем, что при наличии решения достаточно заменить одно число. Пусть это не так,
т.е. найдутся две такие пары, от которых в искомую сумму входят не бóльшие в своих парах числа x1 и y1, а меньшие числа из соответствующих пар: x2 и y2. При этом x2 + y2 имеет остаток от деления на 3, отличный от остатка от деления на 3 числа x1 + y1 (иначе мы могли бы включить в сумму x1 + y1 вместо x2 + y2). Но это означает, что хотя бы одно из чисел x2, y2 тоже при делении на 3 имеет остаток, отличный от соответствующего максимального числа пары. Значит, оптимальной является замена только одного из таких чисел.Программа читает все данные один раз. В каждой паре определяется большее число *Max* и разность между бóльшим и меньшим числами пары *D*. После обработки очередной пары программа хранит два числа: *s* – сумму всех максимальных элементов прочитанных пар и *D_min* – наименьшую возможную разность *D*, не кратную 3. Окончательным ответом будет значение s, если оно не делится на 3, и *s–D_min* в

противном случае. Если *s* делится на 3, а *D_min* не определено (разность между числами во всех парах кратна 3), ответ в соответствии с условиями задачи считается равным 0

**Программа 1.** Пример правильной и эффективной программы для задания Б на языке Паскаль

```
const
    aMax = 10000; {наибольшее возможное число в исходных данных}
var
        N: longint; {количество пар}
        a, b: longint; {пара чисел}
        Max: longint; {максимум в паре}
        Min: longint; {минимум в паре}
        s: longint; {сумма выбранных чисел}
        D_min: longint; {минимальная разница Max-Min не кратная 3}
        i: longint;
begin
        s := 0;
        D_min := aMax + 1;
        readln(N);
        for i := 1 to N do begin
                readln(a, b);
                if a>b then begin Max:=a; Min:=b end
                        else begin Max:=b; Min:=a end;
                s := s + Max;
                if ((Max - Min) mod 3 > 0) and (Max - Min < D_min)
                        then D_min := Max - Min
        end;
        if s mod 3 = 0 then begin
                if D_min > aMax then s := 0
                else s := s – D_min
        end;
        writeln(s)
end.
```

**на языке C++**

```
#include <iostream>
using namespace std;
const int aMax = 10000; //наибольшее возможное число в исходных данных
int main() {
long N; //количество пар
long a, b; //пара чисел
long Max; //максимум в паре
long Min; //минимум в паре
long s; //сумма выбранных чисел
long Dmin; //минимальная разница Max-Min не кратная 3
long i;
s = 0;
Dmin = aMax + 1;
cin >> N;

for (i = 1; i<=N; i++){
        cin>>a>>b;
        if(a > b){
                Max=a; Min=b;
        }else{
                Max=b; Min=a;
        }
        s = s + Max;
        if ((Max - Min) % 3 > 0 && Max - Min < Dmin)
```

```
                Dmin = Max - Min;
}
if (s % 3 == 0){
        if (Dmin > aMax)
                s = 0;
        else
                s = s - Dmin;
}
cout<<s;
return 0;
}
```

Программа 2. Пример правильной и эффективной программы для задания Б на языке Паскаль

```
const
        aMax = 10000; {наибольшее возможное число в исходных данных}
var
        N: longint; {количество пар}
        a: array[1..2] of longint; {пара чисел}
        s_old, s_new: array[0..2] of longint;
        {суммы с соответствующими остатками от деления на 3}
        i, j, k, r: longint;
begin
        readln(N);
        for j := 0 to 2 do
                s_old[j] := 0;
        for i := 1 to N do begin
                readln(a[1], a[2]);
        for j := 0 to 2 do
                s_new[j] := 0;
        for k := 1 to 2 do begin
        for j := 0 to 2 do begin
        if (s_old[j] > 0) or (i = 1) then begin
                r := (s_old[j] + a[k]) mod 3;
        if s_new[r] < s_old[j] + a[k] then
                s_new[r] := s_old[j] + a[k]
        end
        end
        end;
        s_old := s_new
        end;
        if s_new[1] > s_new[2] then
                writeln(s_new[1])
        else
                writeln(s_new[2]);
        {если решения не существует, то s_new[1] и s_new[2]
        окажутся равными нулю}
end.
```

*Замечание для эксперта.* Ученик может «перестраховаться» и явно проверить, что хотя бы одно из чисел *s_new[1], s_new[2]* отлично от 0.
Эта проверка излишня (см. комментарий в конце программы), однако она не влияет на порядок роста времени программы. Снижать баллы за такую избыточную проверку не следует.*Задание А.*
Это задание можно выполнить «в лоб»: сохранить в массиве все исходные данные, перебрать все возможные способы выбора одного элемента из каждой пары и найти максимальную сумму, соответствующую условиям задачи.
*Ниже приводится пример такого решения*

**Решение 2.**

Возможно и решение, основанное на другой идее, а именно будем хранить для каждого прочитанного набора пар три суммы *(s0, s1, s2)* – максимальные суммы элементов пар, имеющие при делении на 3 соответственно остатки 0, 1 и 2. При обработке очередной пары *(a1, a2)* эти суммы обновляются. Для
этого достаточно рассмотреть суммы *s0+a1, s1+a1, s2+a1, s0+a2, s1+a2, s2+a2* и для каждого возможного остатка от деления на 3 выбрать в качестве нового значения *s0, s1 или s2* значение наибольшей из указанных сумм, дающей данный остаток. Окончательным ответом будет бóльшая из сумм *s1 и s2*.

Эта идея приводит к более громоздкой реализации, но все основные требования по эффективности в ней выполнены, поэтому подобное решение при отсутствии ошибок можно оценить максимальным количеством баллов.

Ниже приводится пример основанной на этом принципе программы на языке Паскаль.

*Замечание для эксперта*. В приведённом ниже решении для хранения *s0, s1, s2 используется массив s_new[0..2].* Это упрощает реализацию, однако решение, в котором используются простые переменные, также допустимо. Не следует снижать баллы только за то, что в программе использованы простые
переменные, а не массив, как в приведённом ниже примере

**Пример решения задачи А на языке Паскаль**

```
var
        a: array[1..6, 1..2] of longint;
        i1, i2, i3, i4, i5, i6: longint;
        s, sMax: longint;
begin
        for i1:= 1 to 6 do readln(a[i1,1], a[i1,2]);
                sMax := 0;
        for i1:=1 to 2 do
        for i2:=1 to 2 do
        for i3:=1 to 2 do
        for i4:=1 to 2 do
        for i5:=1 to 2 do
        for i6:=1 to 2 do begin
                s:=a[1,i1]+a[2,i2]+a[3,i3]+a[4,i4]+a[5,i5]+a[6,i6];
                if (s mod 3 <> 0) and (s > sMax) then sMax := s
        end;
        writeln(sMax)
end.
```

---

## Демонстрационный вариант ЕГЭ 2016 г. – задание №27

В физической лаборатории проводится долговременный эксперимент по изучению гравитационного поля Земли. По каналу связи каждую минуту в лабораторию передаётся положительное целое число – текущее показание прибора «Сигма 2015». Количество передаваемых чисел в серии известно и не превышает 10 000. **Все числа не превышают 1000**. Временем, в течение которого происходит передача, можно пренебречь.

Необходимо вычислить «бета-значение» серии показаний прибора – **минимальное чётное произ-**

**ведение двух показаний**, **между моментами передачи которых прошло не менее 6 минут**. Если получить такое произведение не удаётся, ответ считается равным **–1**.

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

*Итоговая оценка выставляется как максимальная из оценок за задания А и Б. Если решение одного из заданий не представлено, то считается, что оценка за это задание – 0 баллов.*

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

**А.** Напишите на любом языке программирования программу для решения поставленной задачи, в которой входные данные будут запоминаться в массиве, после чего будут проверены все возможные пары элементов. Перед программой укажите версию языка программирования.

ОБЯЗАТЕЛЬНО укажите, что программа является решением ЗАДАНИЯ А. Максимальная оценка за выполнение задания А – 2 балла.

**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 времени, если время работы

программы пропорционально количеству полученных показаний прибора N, т.е. при увеличении N в k раз время работы программы должно увеличиваться не более чем в k раз.

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

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. НАПОМИНАЕМ! Не забудьте указать, к какому заданию относится каждая из представленных Вами программ.

Входные данные представлены следующим образом. В первой строке задаётся число N – общее количество показаний прибора. Гарантируется, что N > 6. В каждой из следующих N строк задаётся одно положительное целое число – очередное показание прибора.

*Пример входных данных:*
*11*
*12*
*45*

*5*
*3*
*17*
*23*
*21*
*20*
*19*
*18*
*17*

Программа должна вывести одно число – описанное в условии произведение либо –1, если получить такое произведение не удаётся.

*Пример выходных данных для приведённого выше примера входных данных:*
*54*

**Решение:**

**Задание Б** (решение для задания А приведено ниже, см. программу 4). Чтобы произведение было чётным, хотя бы один сомножитель должен быть чётным, поэтому при поиске подходящих произведений чётные показания прибора можно рассматривать в паре с любыми другими, а нечётные – только с чётными.Для каждого показания с номером k, начиная с k = 7, рассмотрим все допустимые по условиям задачи пары, в которых данное показание получено вторым. Минимальное произведение из всех этих пар будет получено, если первым в паре будет взято минимальное подходящее показание среди всех, полученных от начала приёма и до показания с номером k – 6. Если очередное показание чётное, минимальное среди предыдущих может быть любым, если нечётное – только чётным.Для получения эффективного по времени решения нужно по мере ввода данных помнить абсолютное минимальное и минимальное чётное показание на каждый момент времени, каждое вновь полученное показание умножать на соответствующий ему минимум, имевшийся на 6 элементов ранее, и выбрать минимальное из всех таких произведений.Поскольку каждое текущее минимальное показание используется после ввода ещё 6 элементов и после этого становится ненужным, достаточно хранить только 6 последних минимумов. Для этого можно использовать
массив из 6 элементов и циклически заполнять его по мере ввода данных.
Размер этого массива не зависит от общего количества введённых показаний, поэтому такое решение будет эффективным не только по времени, но и по памяти. Чтобы хранить абсолютный и чётный минимумы, нужно
использовать два таких массива.Ниже приводится пример такой программы, написанной на алгоритмическом
языке.

**Программа 1.** Пример правильной программы на алгоритмическом языке.
*Программа эффективна по времени и по памяти*

```
алг
нач
        цел s = 6 | требуемое расстояние между показаниями
        цел amax = 1001 | больше максимально возможного показания
```

```
        цел N
        ввод N
        цел a | очередное показание прибора
        целтаб мини[0:s-1] | текущие минимумы последних s элементов
        целтаб миничет[0:s-1] | чётные минимумы последних s элементов
        цел i
        | вводим первые s показаний, фиксируем минимумы
        цел ма; ма := amax | минимальное показание
        цел мчет; мчет := amax | минимальное чётное показание
        нц для i от 1 до s
                ввод a
                ма := imin(ма, a)
                если mod(a,2) = 0 то мчет := imin(мчет,a) все
                мини[mod(i, s)] := ма
                миничет[mod(i, s)] := мчет
        кц
        цел мп = amax*amax | минимальное значение произведения
        цел п
        нц для i от s+1 до N
                ввод a
                если mod(a,2)=0
                        то п := a * мини[mod(i, s)]
                        иначе если мчет < amax
                                то п := a * миничет[mod(i, s)]
                                иначе п := amax*amax;
                        все
                все
                мп := imin(мп, п)
                ма := imin(ма, a)
                если mod(a,2) = 0 то мчет := imin(мчет,a) все
                мини[mod(i, s)] := ма
                миничет[mod(i, s)] := мчет
        кц
        если мп = amax*amax то мп:=-1 все
        вывод мп
кон
```

**Программа 2.** *Пример правильной программы.Возможны и другие реализации.*

Например, вместо циклического заполнения массива можно каждый раз **сдвигать его элементы**. В приведённом ниже примере хранятся и сдвигаются не минимумы, а исходные значения.
Это требует чуть меньше памяти (достаточно одного массива вместо двух), но по времени решение со сдвигами менее эффективно, чем с циклическим заполнением. Однако время работы остаётся пропорциональным N, поэтому максимальная оценка за такое решение тоже составляет 4 балла.

Программа использует сдвиги, но эффективна по времени и по памяти

*на языке Паскаль*

```
const s = 6; {требуемое расстояние между показаниями}
amax = 1001; {больше максимально возможного показания}
var
        N: integer;
        a: array[1..s] of integer; {хранение s показаний прибора}
        a_: integer; {ввод очередного показания}
        ma: integer; {минимальное число без s последних}
        me: integer; {минимальное чётное число без s последних}
        mp: integer; {минимальное значение произведения}
        p: integer;
```

```
        i, j: integer;
begin
        readln(N);
        {Ввод первых s чисел}
        for i:=1 to s do readln(a[i]);
        {Ввод остальных значений, поиск минимального произведения}
        ma := amax; me := amax;
        mp :=amax*amax;
        for i := s + 1 to N do begin
                readln(a_);
                if a[1] < ma then ma := a[1];
                if (a[1] mod 2 = 0) and (a[1] < me) then me := a[1];
                if a_ mod 2 = 0 then p := a_ * ma
                else if me < amax then p := a_ * me
                else p := amax* amax;
                if (p < mp) then mp := p;
                {сдвигаем элементы вспомогательного массива влево}
                for j := 1 to s - 1 do
                        a[j] := a[j + 1];
                a[s] := a_
        end;
        if mp = amax*amax then mp:=-1;
        writeln(mp)
end.
```

*на языке C++*

```
#include <iostream>
using namespace std;
const int s = 6;  //требуемое расстояние между показаниями
const int amax = 1001;  //больше максимально возможного показания
int main() {
int N;
int a[s+1]; //хранение s показаний прибора
int a_; //ввод очередного показания
int ma; //минимальное число без s последних
int me; //минимальное чётное число без s последних
int mp; //минимальное значение произведения
int p;
int i, j;

cin>>N;
//Ввод первых s чисел
for (i=1; i<=s; i++)
        cin>>a[i];

//Ввод остальных значений, поиск минимального произведения
ma = amax; me = amax;
mp =amax*amax;

for (i = s + 1; i<=N; i++){
        cin >> a_;
        if(a[1] < ma)
                ma = a[1];
        if(a[1] % 2 == 0 && a[1] < me)
                me = a[1];
        if(a_ % 2 == 0)
                p = a_ * ma;
        else if (me < amax)
                p = a_ * me;
        else
                p = amax* amax;
```

```
        if (p < mp)
                mp = p;
        //сдвигаем элементы вспомогательного массива влево
        for(j = 1;j<=s - 1;j++)
                a[j] = a[j + 1];
        a[s] = a_;
}
if(mp == amax*amax)
        mp=-1;
cout << mp;

return 0;
}
```

Если вместо небольшого массива фиксированного размера (циклического или со сдвигами) хранятся все исходные данные (или все текущие минимумы), программа сохраняет эффективность по времени, но становится неэффективной по памяти, так как требуемая память растёт пропорционально N. Ниже приводится пример такой программы на языке Паскаль. Подобные (и аналогичные по сути) программы оцениваются не выше 3 баллов.

**Программа 3.** Пример правильной программы на языке Паскаль. *Программа эффективна по времени, но неэффективна по памяти*

```
const s = 6; {требуемое расстояние между показаниями}
        amax = 1001; {больше максимально возможного показания}
var
        N, p, i: integer;
        a: array[1..10000] of integer; {все показания прибора}
        ma: integer; {минимальное число без s последних}
        me: integer; {минимальное чётное число без s последних}
        mp: integer; {минимальное значение произведения}
begin
        readln(N);
        {Ввод всех показаний прибора}
        for i:=1 to N do readln(a[i]);
        ma := amax;
        me := amax;
        mp := amax*amax;
        for i := s + 1 to N do
        begin
                if a[i-s] < ma then ma := a[i-s];
                if (a[i-s] mod 2 = 0) and (a[i-s] < me) then
                me := a[i-s];
                if a[i] mod 2 = 0 then p := a[i] * ma
                else if me < amax then p := a[i] * me
                else p := amax * amax;
                if (p < mp) then mp := p
        end;
        if mp = amax*amax then mp := -1;
        writeln(mp)
end.
```

Возможно также переборное решение, в котором находятся произведения всех возможных пар и из них выбирается минимальное. Ниже (см. программу 4) приведён пример подобного решения. Это (и аналогичные ему) решение неэффективно ни по времени, ни по памяти. Оно является решением задания А, но не является решением задания Б. Оценка за такое решение – 2 балла.

**Программа 4.** Пример правильной программы на языке Паскаль. *Программа неэффективна ни по времени, ни по памяти (2б)*

```
const s = 6; {требуемое расстояние между показаниями}
var
        N: integer;
        a: array[1..10000] of integer; {все показания прибора}
        mp: integer; {минимальное значение произведения}
        i, j: integer;
begin
        readln(N);
        {Ввод значений прибора}
        for i:=1 to N do
                readln(a[i]);
        mp := 1000 * 1000 + 1;
        for i := 1 to N-s do begin
                for j := i+s to N do begin
                        if (a[i]*a[j] mod 2 = 0) and (a[i]*a[j] < mp)
                                then mp := a[i]*a[j]
                end;
        end;
        if mp = 1000 * 1000 + 1 then mp := -1;
        writeln(mp)
end.
```

**Программа 5. Пример правильной программы на языке C++.** ***Программа неэффективна ни по времени, ни по памяти (2б)***

```
#include <iostream>
using namespace std;
const int s = 6;  //требуемое расстояние между показаниями
int main() {
        int i, j ,N;
        int a[10000];  //все показания прибора
        int mp;   //минимальное значение произведения
        cin >> N;
        //Ввод значений прибора
        for (i = 0; i<N; i++)
                cin >>  a[i];
        mp = 1000 * 1000 + 1;
        for (i=0; i<N-s; i++)
                for (j=i+s; j<N; j++)
                        if (a[i]*a[j] % 2 == 0 && a[i]*a[j] < mp)
                                mp = a[i]*a[j];
        if (mp == 1000 * 1000 + 1)
                mp = -1;
        cout << mp;
return 0;
}
```

**Программа 6. Пример правильной программы на языке Python.** ***Программа неэффективна ни по времени, ни по памяти (2б)***

```
a = []
n = int(input())
for i in range(0, n):
    a.append(int(input()))
d = 6
```

```
mn = 1000 * 1000 + 1
for i in range(0, n - d):
    for j in range(i + d, n):
        if a[i] * a[j] % 2 == 0 and a[i] * a[j] < mn:
            mn = a[i] * a[j]
if mn == 1000 * 1000 + 1:
    mn = -1
print(mn)
```

---

## ЕГЭ 16.06.2016 по информатике. Основная волна. Вариант 41 (Часть С)

На вход даны пары чисел. Нужно выбрать из каждой пары по одному числу так, чтобы сумма всех выбранных чисел не была кратна 6 и при этом была минимально возможной. Напишите программу, выводящую такую сумму на экран. Если же ее невозможно получить, выведите 0.

*Вам предлагается два задания, связанных с этой задачей: задание А и задание Б. Вы можете решать оба задания или одно из них по своему выбору. Итоговая оценка выставляется как максимальная из оценок за задания А и Б. Если решение одного из заданий не представлено, то считается, что оценка за это задание — 0 баллов.*

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

**А**. Напишите на любом языке программирования программу для решения поставленной задачи, в которой входные данные будут запоминаться в массиве. Перед программой укажите версию языка программирования.

**Обязательно** укажите, что программа является решением **задания А**. Максимальная оценка за выполнение задания А — 2 балла.

**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 времени, если время работы программы пропорционально количеству полученных показаний прибора N, т.е. при увеличении N в k раз время работы программы должно увеличиваться не более чем в k раз. Программа считается эффективной по памяти, если размер памяти, использованной в программе для хранения данных, не зависит от числа N и не превышает 1 килобайта.

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

27. Создание программы для анализа числовых последовательностей памяти, — 3 балла.

Напоминаем! Не забудьте указать, к какому заданию относится каждая из представленных Вами программ.

Задача А. Количество пар известно заранее и равно 6. Числа не превышают 30 000.

Пример входных данных:

5 4

3 2

1 1

18 3

11 12

2 5

Пример выходных данных:

23

Задача Б. Количество пар N не известно заранее и может принимать значения 2 <= N <= 200 000. На вход подается сначала количество пар, затем сами пары. Числа по модулю не превышают 30 000.

Пример входных данных:

6

5 4

3 2

1 1

18 3

11 12

2 5

Пример выходных данных:

23

**Решение:**

Задание Б

```
n=int(input())
mn1=0 #минимальная сумма
r=30000*2 #минимальная разница между парой элементов
mn2=0 # сложение минимальной разиницы (r) и минимальной суммы
a=0:b=0
for i in range(0,n):
        a,b=map(int,input().split())
        if a<b:
                mn1+=a
        else:
                mn1+=b
        if abs(a-b)<r and abs(a-b)%6!=0 and a!=b:
                r=abs(a-b)
if mn1%6==0 and r=30000*2:
        print(0)
elif mn1%6==0:
        mn2=mn1+r
        print(mn2)
else:
        print(mn1)
```

---

В некотором вузе абитуриенты проходили предварительное тестирование, по результатам которого они могут быть допущены к сдаче вступительных экзаменов в первом потоке. Тестирование проводится по трём предметам, по каждому предмету абитуриент может набрать от 0 100 баллов. При этом к сдаче экзаменов в первом потоке допускаются абитуриенты, набравшие по результатам тестирования не менее 30 баллов по каждому из трёх предметов, причём сумма баллов должна быть не менее 140. На вход программы подаются сведения о результатах предварительного тестирования. Известно, что общее количество участников тестирования не превосходит 500.

В первой строке вводится количество абитуриентов, принимавших участие в тестировании, **N**. Далее следуют **N** строк, имеющих следующий формат:

**<Фамилия> <Имя> <Баллы>**

Здесь *<Фамилия>* – строка, состоящая не более чем из 20 символов; *<Имя>* – строка, состоящая не более чем из 15 символов, *<Баллы>* – строка, содержащая два целых числа, разделенных пробелом – баллы, полученные на тестировании по каждому из трёх предметов. При этом *<Фамилия>* и *<Имя>*, *<Имя>* и *<Баллы>* разделены одним пробелом. Пример входной строки:

**Романов Вельямин 48 39 55**

Напишите программу, которая будет выводить на экран фамилии и имена абитуриентов, допущенных к сдаче экзаменов в первом потоке. При этом фамилии должны выводиться в

27. Создание программы для анализа числовых последовательностей алфавитном порядке.

**Решение:**

```
#include <iostream>
#include <string>
using namespace std;
int main(){
        int i, j, N, mark1, mark2, mark3, count=-1;
        string Result[500];
        string name, surname, temp;
        cin>>N;
        for(i=0; i<N; i++){
                cin>>name>>surname>>mark1>>mark2>>mark3;
                if(mark1>=30 && mark2>=30 && mark3>=30 && mark1+mark2+mark3>=140){
                        count++;
                        Result[count]= name + " " + surname;
                }
        }
        for(i=0; i<=count-1; i++){
                for(j=i+1; j<=count; j++)
                        if(Result[i]>Result[j]){
                                temp=Result[i];
                                Result[i]=Result[j];
                                Result[j]=temp;
                        }
        }
        for(i=0;i<=count;i++)
                cout<<Result[i]<<endl;
return 0;
}
```

---

В соревнованиях по многоборью (из M видов спорта) участвуют N спортсменов (N < 1000) . На вход программе в первой строке подается число спортсменов N, во второй – число видов спорта M. В каждой из последующих N строк находится информация в следующем формате:

**<Фамилия> <Имя> <Баллы>**

где <*Фамилия*> – строка, состоящая не более, чем из 20 символов без пробелов, <*Имя*> – строка, состоящая не более, чем из 12 символов без пробелов, <*Баллы*> – M целых чисел, обозначающие количество баллов, набранных спортсменом в каждом из видов многоборья.

<*Фамилия*> и <*Имя*>, <Имя> и <Баллы>, а также отдельные числа в поле <*Баллы*> разделены ровно одним пробелом. Пример входных строк:

**3**

**4**

**Иванов Сергей 100 30 78 13**

**Петров Антон 90 16 98 14**

**Сидоров Юрий 100 70 30 21**

Программа должна выводить результирующую таблицу, содержащую список спортсменов, отсортированный по убыванию суммы баллов, набранные суммы и занятые места.

В данном случае программа должна вывести

**Иванов Сергей 221 1**

**Сидоров Юрий 221 1**

**Петров Антон 218 2**

**Решение:**

```
#include <iostream>
#include <string>
using namespace std;
struct Tns{
        string namesurname;
        int sum;
};
int main(){
        Tns ns[1000];
        int N, M, i, j, mark, mesto;
        Tns temp;
        string name, surname;
        cin>>N>>M;
        for (i=0;i<N; i++){
                cin>>name>>surname;
                ns[i].namesurname = name + " " + surname;
                ns[i].sum=0;
                for(j=0; j<M; j++){
                        cin>>mark;
                        ns[i].sum += mark;
                }
        }
        for(i=0; i<N-1; i++){
                for(j=i+1; j<N; j++)
                        if(ns[i].sum<ns[j].sum){
                                temp=ns[i];
                                ns[i]=ns[j];
                                ns[j]=temp;
                        }
        }
        mesto=1;
        for(i=0; i<N ;i++){
                if(i>0 && ns[i].sum<ns[i-1].sum)
                        mesto++;
                cout<<ns[i].namesurname<<" "<<ns[i].sum<<" "<<mesto<<endl;
        }
return 0;
}
```

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 на 3 позиции: буква «E» превращается в букву «H», буква «t» — в букву «w» буква «Y» — в букву «B».

Напишите эффективную, в том числе и по используемой памяти, программу (укажите используемую версию языка программирования, например Borland Pascal 7.0), которая должна закодировать строку по указанному алгоритму.

На вход программе подается 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, qmgi. «Ciev» — b tpzahrl#

**Решение:**

```
#include <fstream>
#include <string>
using namespace std;
int main() {
        ifstream fin("input.txt"); ofstream fout("output.txt");
        string s;

        while (!fin.eof()) {
                fin >> s;
                int k = 0;
```

```
                for (int i = 0; i < s.length(); i++)
                        if ((s[i] >= 'a' && s[i] <= 'z') || (s[i] >= 'A' && s[i] <= 'Z'))
                                k++;
                for (int i = 0; i < s.length(); i++) {
                        if (s[i] >= 'a' && s[i] <= 'z')
                                if ((int)s[i] + k <= (int)'z')
                                        fout << char((int)s[i]+ k);
                                else
                                        fout << char((int)'a'+ k - ((int)'z'-(int)s[i])-1);
                        else if(s[i] >= 'A' && s[i] <= 'Z')
                                        if ((int)s[i] + k <= (int)'Z')
                                                fout << char((int)s[i] + k);
                                        else
                                                fout << char((int)'A' + k - ((int)'Z' -
(int)s[i]) - 1);
                        else
                                fout << s[i];
                }
                fout << " ";
        }



        return 0;
}
```