

**Заключительный этап академического соревнования Олимпиады школьников
«Шаг в будущее» по программированию «Профессор Лебедев»
(общеобразовательный предмет информатика), весна 2021 год**

10 класс

Вариант 2

Задача 1

Интернет-тролль Ибрагим просыпается в h часов m минут и сразу же начинает без остановки строчить сообщения в соцсети, провоцируя пользователей на эмоциональную перепалку с переходом на личности. На написание и отправку каждого сообщения уходит n секунд.

Юзер Юся просыпается ровно в 8 часов, и тут же начинает потреблять сегодняшние сообщения, тратя по 30 секунд на каждое. Если сообщение тролля уже написано, потребляет именно его.

Устойчивость юзера Юси к провокациям составляет k сообщений тролля, после прочтения которых юзер ввязывается в перепалку тем, что тут же после прочтения k -го сообщения тролля пишет первое сообщение сам.

Временем начала перепалки считается время начала написания юзером первого сообщения. Сколько времени будет на часах, когда интернет-тролль Ибрагим достигнет своей цели?

На вход программе в первой строке через пробел подаются 4 целых числа – данные тролля: h, m, n и k . Время, когда просыпаются тролль и юзер, указано в одних сутках. $n, k > 0$. Исходные данные заданы так, что время начала перепалки гарантированно будет в тех же сутках.

В одной строке выходных данных через пробел программа должна вывести искомое время: час, минуту и секунду, которые будут на часах в момент начала перепалки.

Пример

Входные данные	Выходные данные
8 0 30 4	8 2 30

Комментарий к примеру

Время начал написания сообщений: 8:00:00, 8:00:30, 8:01:00, 8:01:30.

Время, когда юзер будет начинать читать сообщения: 8:00:30, 8:01:00, 8:01:30, 8:02:00, после чего в 8:02:30 начнёт писать своё.

Тесты

Входные данные	Выходные данные
8 0 30 4	8 2 30
8 0 60 4	8 4 30
4 0 10 1000	16 20 0
8 0 10 1000	16 20 30
8 10 10 1000	16 30 30
17 10 10 800	23 50 30
8 0 60 10	8 10 30
7 0 60 10	8 5 0
7 55 60 10	8 5 30
9 0 35 7	9 5 0

Решение

```
hi, mi, ni, k = map(int, input().split())
```

```
hu, mu, nu = 8, 0, 30
```

```
i = (hi * 60 + mi) * 60
```

```
u = (hu * 60 + mu) * 60
```

```
# время окончания написания k сообщений
```

```
ik = i + ni * k
```

```

# время окончания написания троллем первого сообщения
i1 = i + ni
# если юзер проснулся не раньше возможности прочитать первое сообщение тролля
if u >= i1:
    # временем начала чтения является время, когда юзер проснулся
    start = u
else:
    # временем начала чтения является время окончания чтения после написания
    start = (i1 - u + nu - 1) // nu * nu + u
# время, когда юзер мог бы прочитать сообщения, если бы они были уже написаны
finish = start + k * nu
# если чтение было подряд (время окончания написания k сообщений не позже начала чтения
последнего сообщения), то
if ik <= finish - nu:
    res = finish
else:
    # временем начала чтения последнего сообщения
    start = (ik - u + nu - 1) // nu * nu + u
    res = start + nu
s = res % 60; res //= 60 # переходим в минуты
m = res % 60; res //= 60 # переходим в часы
h = res % 24; res //= 24 # переходим в дни
print(h, m, s)

```

Задача 2

В небоскребе n этажей ($n < 10^{10}$). Известно, что если уронить некий стеклянный шарик с этажа номер p , и шарик не разобьется, то если уронить шарик с этажа номер $p-1$, то он тоже не разобьется. Известно также, что если уронить стеклянный шарик с этажа номер p , и шарик разобьется, то если уронить шарик с этажа номер $p+1$, то он тоже разобьется. Кроме того известно, что при броске с последнего этажа шарик всегда разбивается.

Вы хотите определить минимальный номер этажа, при падении с которого шарик разбивается. Для проведения экспериментов у вас есть два шарика. Вы можете разбить их все, но в результате вы должны абсолютно точно определить этот номер.

Определите, какого числа бросков достаточно, чтобы заведомо решить эту задачу.

Формат входных данных

Программа получает на вход натуральное число – количество этажей в небоскребе.

Формат выходных данных

Требуется вывести наименьшее число бросков, при котором можно всегда решить задачу.

Пример

Входные данные	Выходные данные
2	1

Тесты

Входные данные	Выходные данные
1	0
2	1
10	4

100	14
93	14
92	13
800000000	40000

Решение (C++)

```
#include <iostream>
#include <cmath>

int main(void)
{
    int n = 0;
    std::cin >> n;
    std::cout << ceil((-1 + sqrt(1 + 8 * (n - 1))) / 2);
    return 0;
}
```

Задача 3

Автомат получает на вход натуральное число X (не превышающее 10^6). По этому числу строится новое *шестнадцатеричное* число Y :

- первая цифра нового числа – это крайняя правая цифра двоичного представления X ;
- вторая цифра числа Y – это крайняя правая цифра четверичного представления X ;
- третья цифра числа Y – это крайняя правая цифра восьмеричного представления X .

Далее цифры выстраиваются по неубыванию, а затем полученное шестнадцатеричное число переводится в десятичное - результат работы автомата число Y .

Например, $X = 187_{10} = 10111011_2 = 2323_4 = 273_8$. Тогда новое число $Y = 133_{16} = 307_{10}$.

Напишите программу, которая будет искать наибольшее из чисел заданного интервала $[A, B]$, на котором автоматом будет получено наименьшее число, а также сколько раз на данном интервале будет выдано наименьшее значение.

На вход программы подаётся два целых числа A и B ($1 \leq A \leq B \leq 1\,000\,000$), записанных через пробел.

Программа должна вывести в одной строке через пробел два целых числа - вычисленные значения.

Входные данные	Вывод	Примечание
185 187	186 1	185 -> 273 186 -> 34 наименьшее 187 -> 307

Тесты

Ввод	Вывод
1 10	8 1
159 159	159 1
1968 1980	1976 2
1 1000000	1000000 125000
999900 1000000	1000000 13

Решение

```
program automatic1;
```

```
function automatic(x: longint): longint;
```

```
var digit1, digit2, digit3: integer;
```

```
begin
```

```
    digit3 := x mod 8;
```

```
    digit2 := x mod 4;
```

```
    digit1 := x mod 2;
```

```
    automatic := digit1 * 256 + digit2 * 16 + digit3
```

```
end;
```

```
var a, b, n, min, y, xmin, k: longint;
```

```
begin
```

```
    readln(a, b);
```

```
    min := 100000;
```

```
    k := 0;
```

```
    for n := b downto a do
```

```
        begin
```

```
            y := automatic(n);
```

```
            if y < min then
```

```
                begin
```

```
                    min := y;
```

```
                    xmin := n;
```

```
                    k := 1
```

```
                end
```

```
            else
```

```
                if y = min then
```

```
                    k := 1 + k;
```

```
        end;
```

```
        write(xmin, ' ', k)
```

```
end
```

Задача 4

Шарики массой 100 г бросают в воду. Будем называть эталонным шариком тот, чей объем ближе всего к среднему арифметическому объемов тонущих в воде шариков. Зная объем каждого шарика, найти номер эталонного. Гарантируется, что среди шариков есть хотя бы один тяжелее воды.

Формат ввода

В следующей строке вводится натуральное число N ($1 \leq N \leq 15$) – количество шариков. Далее в N строчках вводится по одному натуральному числу v_i – объем шарика под номером i . Объем указывается в кубических сантиметрах и не превышает 1000 см^3 . Плотность воды – 1 г/см^3 .

Формат вывода

Вывести через пробел одно целое число – номер эталонного шарика. Если таких шариков несколько, вывести шарик с меньшим номером.

Пример

Входные данные	Выходные данные
5 75 102 98	1

54 105	
-----------	--

Тесты

Входные данные	Выходные данные
5 75 102 98 54 105	1
1 76	1
10 100 100 100 95 95 103 95 100 100 100	4
4 100 5 6 7	3
9 7 9 9 9 9 100 11 105 100	2

Решение

```
program mzv2;
```

```
const
```

```
  K=15;
```

```
type
```

```
  tmas = array [1..K] of integer;
```

```
var
```

```
  mas:tmas;
```

```
  sr,min:real;
```

```

m,sum,kol,n,i,mi:integer;
begin
readln(n);
sum:=0;
kol:=0;
for i:=1 to n do
  begin
    readln(mas[i]);
    if mas[i]>100 then
      begin
        sum:=sum+mas[i];
        kol:=kol+1;
      end;
    end;
  min:=1001;
  sr:=sum/kol;
  for i:=1 to n do
    if abs(mas[i]-sr)<min then
      begin
        min := abs(mas[i]-sr);
        mi:=i;
      end;
  writeln(mi);
end.

program mzv1;

const
  K=15;
type
  tmas = array [1..K] of integer;

var
  mas:tmas;
  sr,min:real;
  m,sum,kol,n,i,mi:integer;
begin
readln(n);
sum:=0;
kol:=0;
for i:=1 to n do
  begin
    readln(mas[i]);
    if mas[i]>100 then
      begin
        sum:=sum+mas[i];
        kol:=kol+1;
      end;
    end;
  min:=1001;
  sr:=sum/kol;
  for i:=1 to n do

```

```

if abs(mas[i]-sr)<min then
  begin
    min := abs(mas[i]-sr);
    mi:=i;
  end;
writeln(mi);
end.

```

Задача 5

Антон, Борис и Владимир передают зашифрованные сообщения в Центр. Сообщения состоят только из строчных латинских букв. В каждом сообщении последовательность букв **qvb** встречается только один раз и является разделителем. В сообщении от Антона в цепочке знаков от начала до разделителя на одну букву больше, чем от разделителя до окончания, от Бориса – на две, а от Владимира – ровно на три. В остальных случаях это сообщение считается шумом. Найдите, кто отправил больше всего сообщений. Такой агент обязательно будет один. Код Антона – 1, код Бориса – 2, код Владимира – 3.

Формат ввода

В первой строке вводится сначала натуральное число n ($n \leq 1000$) – количество сообщений, затем в n следующих строках записано по одному сообщению.

Формат вывода

Вывести одно целое число – код агента, который передал больше всего сообщений в Центр.

Примеры

Входные данные	Выходные данные
3 abcqvbf bcfqvbe abqvbr	1

Тесты

Входные данные	Выходные данные
3 abcqvbf bcfqvbe abqvbr	1
7 avqqqvbd aqvb aqvb aaqv aaqv aqvb avqqqvbd	1
7 aaaqv aaaqv ayaqvbuia ayutqv qvba	3

eaqvb ₁ braa y ₁ yy ₁ qvbyay	
5 qvb qvb qvba qvba aaaqvb	3
1 aqvb	1

Решение

program szv3;

function delta(s,s1:string):integer;

var

l:integer;

begin

l:=length(copy(s,1,pos(s1,s)-1));

delete(s,1,pos(s1,s)+length(s1)-1);

delta:=1 - length(s);

end;

var

n,i,a,b,c,k:integer;

s:string;

begin

readln(n);

a:=0;

b:=0;

c:=0;

for i:=1 **to** n **do**

begin

readln(s);

if delta(s,'qvb')=1 **then**

a:=a+1;

if delta(s,'qvb')=2 **then**

b:=b+1;

if delta(s,'qvb')=3 **then**

c:=c+1;

end;

if (a>b) **and** (a>c) **then**

writeln(1);

if (b>a) **and** (b>c) **then**

writeln(2);

if (c>a) **and** (c>b) **then**

writeln(3);

end.

Задача 6 Ситуационная задача

Котел, в котором идет химическая реакция, накрыт легкой прочной крышкой, которая не проваливается внутрь. Для нормального протекания реакции необходимо, чтобы выделяющийся пар в течение всего процесса не мог сорвать крышку с котла. Предельное давление пара известно и составляет P . На крышку можно поместить два груза, чтобы пар не мог сорвать крышку. Зная массы грузов и площадь крышки котла, определите, какую пару грузов с наименьшей суммарной массой надо положить на котел, чтобы крышка осталась на месте. Считать, что пар давит на крышку равномерно. Гарантируется, что подходящая пара грузов существует.

Формат ввода

В первой строке программы вводится вещественное положительное число S – площадь крышки котла. Во второй строке программы вводится положительное вещественное число P – предельное давление пара. В третьей строке вводится натуральное число N ($2 \leq N \leq 15$) – количество грузов. Далее в N строчках вводится по одному положительному вещественному числу m_i – масса груза под номером i . Все величины указываются в системе СИ, имеют явный физический смысл и не превышают 10000. Ускорение свободного падения $g = 10 \text{ м/с}^2$.

Формат вывода

Вывести через пробел два целых числа – номера пары грузов с наименьшей суммарной массой, которые надо положить на котел, чтобы крышка осталась на месте. Требуется вывести номера грузов в порядке возрастания. Если таких пар несколько, вывести пару с меньшим минимальным номером и как можно меньшей суммой номеров.

Пример

Входные данные	Выходные данные
1.5 100.5 4 5.5 10.2 7.5 8.6	1 2

Тесты

Входные данные	Выходные данные
1.5 100.5 4 5.5 10.2 7.5 8.6	1 2
1 100 4 5 6 1 9	3 4
10 10 5 5 6 5.5	3 5

8 4.9	
10 10 3 1 5 5	2 3
10 10 7 5 5 6 7 8 5 5	1 2

Решение

```
program s10zv2;
```

```
const
```

```
K = 15;
```

```
G = 10;
```

```
type
```

```
tmas = array [1..K] of real;
```

```
var
```

```
mas:tmas;
```

```
s,p,min:real;
```

```
n,i,j,mi,mj:integer;
```

```
begin
```

```
readln(s);
```

```
readln(p);
```

```
readln(n);
```

```
mi:=K+1;
```

```
mj:=K+1;
```

```
for i:=1 to n do
```

```
  begin
```

```
    readln(mas[i]);
```

```
    min:=min+mas[i];
```

```
  end;
```

```
min:=min*s;
```

```
for i:=1 to n-1 do
```

```
  for j:=i+1 to n do
```

```
    begin
```

```
      if ((mas[i]+mas[j])*G - p*s >= 0) and (mas[i]+mas[j] < min) then
```

```
        begin
```

```
          mi:=i;
```

```
          mj:=j;
```

```
          min:=mas[i]+mas[j];
```

```
        end;
```

```
      end;
```

```
writeln(mi,' ',mj)
```

```
end.
```