Задача №4.

На каждой клетке игрового поля размером $M \times N$ клеток можно поместить либо дозорную башню с лучниками, либо стену, либо оставить его пустым. Лучники на башне, находящейся по координатам (x, y), могут атаковать другие башни по координатам (x_{-2}, y_{-2}), (x_{-2}, y_2), (x_2, y_{-2}) и (x_2, y_2), в том случае, когда они не разделены стеной.

На вход подается карта, на который отмечены свободные поля и поля с размещенными на них объектами. Ваша задача – разместить на карте как можно больше дополнительных башен, лучники на которых ***не будут*** атаковать друг друга.

как понимать x_2 и y_2? Это x_{+2} и y_{+2} или какие то другие координаты?

Входные данные:

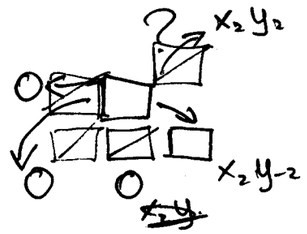
На вход может быть подано несколько тестов, каждый из которых имеет следующую структуру:

- Первая строка содержит два положительных целых числа M и N, разделенных пробелом
- Следующие M строк содержат N символов без пробелов, описывающих игровое поле, где
  - F – свободное поле
  - G – поле с башней
  - P – поле со стеной

Ввод данных считается законченным, когда M и N равны 0.

Выходные данные:

Целые числа, соответствующие количеству дополнительных башен.

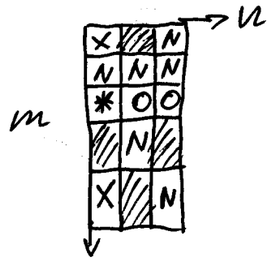


Пример:

тут должно быть 3 или 4?

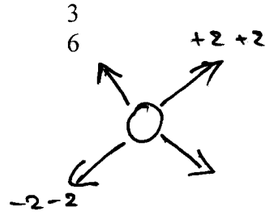
Ввод:

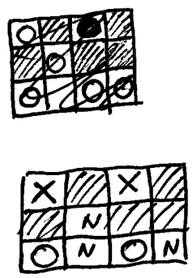
```
FPFP
PFPP
GFGF
5 3
FPF
FFF
FGG
PFP
FPF
0 0
```



Вывод:

```
3
6
```





****

Требования к оформлению задач по программированию:

1) *Программы должны быть написаны на одном из языков: C, C++, Pascal*
2) *Полностью оформленная задача должна содержать:*
   - *программу, выполняющую необходимые операции для всех допустимых данных;*
   - *операции с файлами входных и выходных данных **или** понятный пользователю интерфейс ввода исходных данных;*
   - *комментарии к тексту программы, облегчающие ее понимание.*

Невыполнение вышеуказанных требований влечет за собой снижение получаемых за задачи баллов

см. лист!

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

82

1	2	3	4	Σ
6	4	12	19	41

заполняется жюри!

ПИСЬМЕННАЯ РАБОТА УЧАСТНИКА ОЛИМПИАДЫ ШКОЛЬНИКОВ СПбГУ 2016–2017

заключительный этап

Предмет (комплекс предметов) Олимпиады ИНФОРМАТИКА (10-11 КЛАССЫ)

Город, в котором проводится Олимпиада Санкт-Петербург

Дата 11 марта 2017

****

Вариант 10

****

Задача №1.

На планете «Лютые лютики» имеется свой набор символов для использования в именах жителей. Алфавит состоит из $F > 10$ символов. Имена всех жителей планеты состоят из $D > 4$ букв. Правитель планеты требует, чтобы кто-нибудь смог создать таблицу всех возможных имен, при учете, что никакая буква в имени не повторяется более или равно 4 раза.

Входные данные: Размер алфавита, Алфавит строкой, длина имени

Выходные данные: строки имен (в файл или на экран).

****

Требования к оформлению задач по программированию:

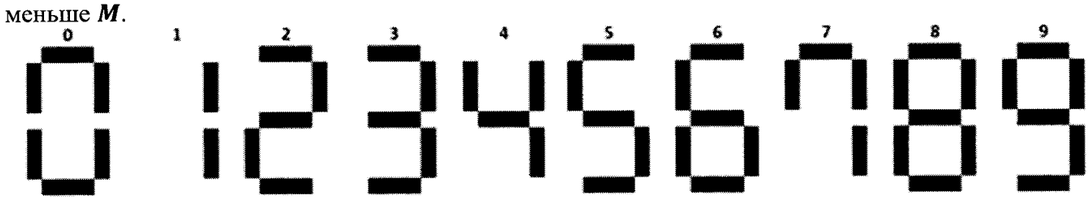
3) *Программы должны быть написаны на одном из языков: C, C++, Pascal*
4) *Полностью оформленная задача должна содержать:*
   - *программу, выполняющую необходимые операции для всех допустимых данных;*
   - *операции с файлами входных и выходных данных **или** понятный пользователю интерфейс ввода исходных данных;*
   - *комментарии к тексту программы, облегчающие ее понимание.*

Невыполнение вышеуказанных требований влечет за собой снижение получаемых за задачи баллов

см. лист!

Задача №2.

Цифровое табло, состоящее из семи-сегментных дисплеев, используется для вывода числовых значений. Вы можете ***деактивировать уже включенные*** сегменты дисплея, но ***включать*** выключенные сегменты не можете. Необходимо написать программу или алгоритм на языках C, C++, Pascal для определения минимального числа, которое может быть выведено на дисплей и не будет меньше ***M***.



Входные данные:

Целое число и ограничение M. Количество дисплеев равно количеству цифр во введенном числе.

Выходные данные:

Целое число. Количество цифр в конечном варианте должно быть равно количеству цифр в начальном.

Здесь ошибка? Ведь чтобы вывести минимальное число ≥10, я могу погасить первые 8 дисплеев, а на последних двух вывести „1". Тогда число 11≤15, но ≥10 => 15 - не минимальное!

Пример:

2 цифры

Ввод: 86 10

M

Вывод: 15

****

Требования к оформлению задач по программированию:

5) *Программы должны быть написаны на одном из языков: C, C++, Pascal*
6) *Полностью оформленная задача должна содержать:*
   - *программу, выполняющую необходимые операции для всех допустимых данных;*
   - *операции с файлами входных и выходных данных **или** понятный пользователю интерфейс ввода исходных данных;*
   - *комментарии к тексту программы, облегчающие ее понимание.*

Невыполнение вышеуказанных требований влечет за собой снижение получаемых за задачи баллов

см. лист!

Я считаю, что для вывода числа используются сначала все дисплеи справа; если их не хватает по разрядности, добавляется еще один слева и т.д.

Я понимаю, что ~~это~~ первое число на вводе нужно для определения числа дисплеев, но ~~в~~ для моей реализации оно не нужно.
(функция подсчета количества цифр в числе, если что, есть на черновике).

4

Задача №3.

Существует такая комната прямоугольной формы, с целочисленными длиной **l** и шириной **w**, для которых верно равенство **l = Aw + B**, где **A** и **B** – целочисленные постоянные. Количество единиц площади этой комнаты таково, что если прибавить к нему **C**, то результат будет делиться без остатка на простое число **P**. Найдите все возможные значения **w**.

Входные данные:

Первая строка содержит количество тестовых случаев, каждая следующая – тестовый случай с разделенными пробелом значениями **A**, **B**, **C** и **P**.

Выходные данные:

Для каждого случая с новой строки выводится результат решения, который содержит общее количество решений и полученные значения ширины в порядке возрастания. Все значения разделяются пробелом

Ограничения:

- $2 \le P < 10^6$
- $0 < A < P$
- $0 \le B$
- $C < P$

Пример:

Ввод:	***Вывод:***
2 1 1 0 2 1 2 2 3	2 0 1 0

****

Требования к оформлению задач по программированию:

7) *Программы должны быть написаны на одном из языков: C, C++, Pascal*
8) *Полностью оформленная задача должна содержать:*
   - *программу, выполняющую необходимые операции для всех допустимых данных;*
   - *операции с файлами входных и выходных данных **или** понятный пользователю интерфейс ввода исходных данных;*
   - *комментарии к тексту программы, облегчающие ее понимание.*

Невыполнение вышеуказанных требований влечет за собой снижение получаемых за задачи баллов

см. лист!

(2) C++ Чистовик

```
#include <iostream>;
#include <vector>;
#include "math.h";
using namespace std;
vector <vector<int>> a = new vector<vector<int>>(); // здесь i - разрядность числа
int makeNum(int i, int res){
    for(int c=0; c<a[i].size(); c++){ // пробег по цифрам
        if (res+ a[i][c]*pow(10;i) >=M){ // предыдущее число+
            return res+a[i][c]*pow(10;i);} // новый разряд.
        else{ makeNum(i+1, res);}
                }
    // если ни один из разрядов не подошел, тогда еще +1 разряд
}
int main(){ // инициализация массива.
    int n, m, res;
    cin>>n>>m;
    res= makeNum(0, 0);
    cout<< res <<endl;
return 0;
}
```

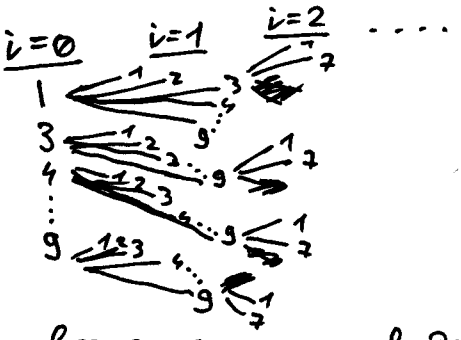
```
a[0]= new vector<int>[1, 3, 4, 5, 7, 9];
a[1]= new vector<int>[0,1, 2,3,4,5,6,7,8,9];
a[2]= new vector<int>[1,7];
a[3]= new vector<int>[5,6];
a[4]= new vector<int>[5];
a[5]= new vector<int>[1,4];
a[6]= new vector<int>[1,3];
a[7]= new vector<int>[2];
a[8]= new vector<int>[1];
a[9]= new vector<int>[1,7];
```

Комментарий:

1) Массив (вектор) a – хранит все возможные варианты цифр для каждого элемента, отсортированные по возрастанию.

2) Функция makeNum – рекурсивная функция, строящая все возможные комбинации из цифр массива a и получающая все возможные числа на табло в порядке возрастания.
Сначала перебирает единичные разряды, потом десятки и т.д... Пример рекурсия необходима, чтобы сохранять предыдущие разряды.

Графика:



Т.к. идем по возрастанию, заведомо знаем, что первый полученный результат – минимальный.

① C++ Числовик

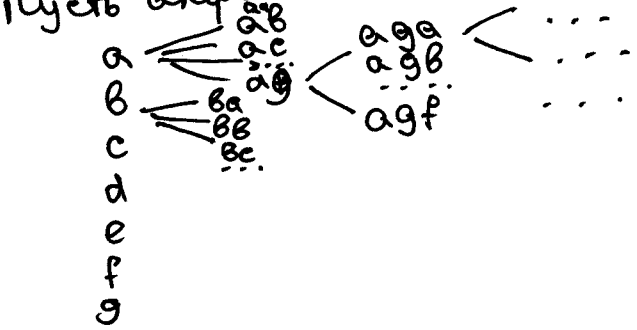
```
#include <iostream>
#include <string>
using namespace std;
string S; int d;
void makeName (vector<int> kolv, string name, int i); //объявим функцию, //т.к. она реализована //-на ниже чем main.

int main(){ int f,
    cin>>f;
    cin>>S; vector <int> kolv=new vector<int>(f);
    cin>>d;
      for(int i=0; i<f; i++){          //инициализация вектора.
            kolv[i]=0;
                            }
     for (int i=0; i<f; i++){ kolv[i]++; //вызываем ф-ию makeName
          makeName(kolv, S[i], 1);  от каждой из букв алфавита.
          kolv[i]--;
                            }
  return 0;
}
void makeName (vector<int> kolv, string name, int i){
   if (i>=d) cout<<name; //Выводим имена из 4х и более букв.
   for(int j=0; j<kolv.size(); j++){ //пробегаем по алфавиту
       if (kolv[j]<2) { kolv[j]++; newName=name+S[j]; //для каждой
                        makeName(kolv, newName, i+1); //буквы алфавита
                        kolv[j]--;                    //добавляем её в
                                                      // конец имени и
                                                      //идем дальше.
                      }
                    }
                  }
```

Комментарий:

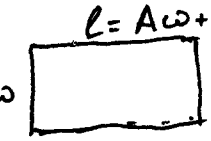
1) Функция makeName – рекурсивная фция, генерирующая все возможные имена, подходящие под условия(длина, кол-во повторяющихся букв).
Вызывается от каждой буквы алфавита, далее на каждой ступени рекурсии добавляет в конец имени все буквы алфавита.

Графика:
Пусть алфавит: „abcdefg".



Числовик

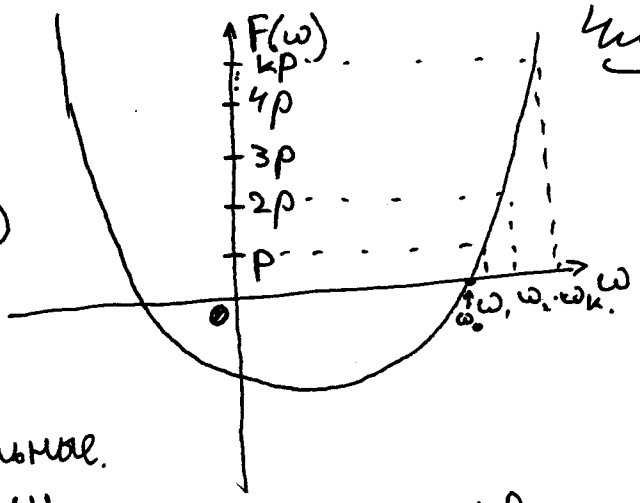
(3)



$S = \omega(A\omega + B)$

По условию: $\omega(A\omega+B) + C \vdots p$ (1)

$F(\omega) = \underbrace{A\omega^2 + B\omega + C}_{\text{квадратная функция.}} \vdots p$



Считаем, что A, B, C, P – натуральные.

1) ~~Сначала проверим, делится ли~~ ~~(и перебор?)~~ $A\omega^2 + B\omega + C = kp$.

Вопрос: для каких k $\exists\, \omega$: Нужно ограничить перебор по k! Это задача по математике!

Комментарий:

1) $A\omega^2 + B\omega + C \vdots p \Leftrightarrow A\omega^2 + B\omega + C = kp$, где $k \in \mathbb{Z}$

Тогда будем увеличивать k и проверять корни уравнения $A\omega^2 + B\omega + (C - kp) = 0$ ~~на целость~~ на целость? ($\omega_1, \omega_2 \in \mathbb{Z}$).

C++

```
#include <iostream>;
#include <math.h>;
#include <vector.h>
using namespace std;
bool condition(int k);
int main(){
    int kol;
    cin >> kol;
    long double [] a = new long double [k];
    long double [] b = new long double [k];
    long double [] c = new long double [k];
    long double [] p = new long double [k];
    for (int i=0; i<k; i++){
        cin >> a[i] >> b[i] >> c[i] >> d[i];
    }
    for (int i=0; i<k; i++){ int k=0 ; vector <long long> roots = new vector<long long>();
    while (condition(k)){ // пока выполняется ограничение на k.
    long double desc = b*b - 4*a*(c-k*p);
        if(desc>=0){desc = sqrt(desc);} // дискриминант
        else break;
long double root1 = (-b+desc)/(2*a); // корни ур-я
long double root2 = (-b-desc)/(2*a);
        if (root1 - (long long)root1 <=0) roots.pushback((long long)root1); // приведем тип
        if (root2 - (long long)root2 <=0) roots.pushback((long long)root2); // проверим, целые ли, // если да, добавим в ответ

    cout << roots.size();
        for (int i=0; i<roots.size(); i++) cout << " " << roots[i];

}
bool condition(int k){
        return k<1000;}.
```

(4) C++. Числовик

```
#include <iostream>
#include <math.h>
#include <vector>
using namespace std; void markB(int i, int j); //реализована ниже main

int main(){
    int m, n;
    cin >> m >> n; int res=0;                                   //ввод
    vector<vector<string>> a = new vector<vector<string>>();
    for (int i=0; i<m; i++){
        string s; cin >> s; a[i]= new vector<string>();
        for (int k=0; k<s.length; k++)
            a[k] = s[i];
    }
    for (int i=0; i<m; i++){                                    //пометим точки, находящиеся под
        for (int j=0; j<n; j++){                                // прицелом данных башен.
            if( a[i][j]=="G"){ markB(i,j);}
    }}
    for(int i=0; i<m; i++){                                     //если поле осталось
        for(int j=0; j<n; j++){                                 свободно, сделаем его
            if(a[i][j]=="F"){ a[i][j]="G1"; res++;              башней и отметим
                markB(i;j);                                     точки, находящиеся у
            }                                                   неё под ударом.
    cout << res;
    return 0;
}
    void markB (int i, int j){
        if (i-2 >=0 && j-2>=0 && a[i-1][j-1]!="P") a[i-2][j-2]="B";
        if (i+2 <m && j+2<n && a[i+1][j+1]!="P") a[i+2][j+2]="B";
        if (i-2 >=0 && j+2<n && a[i-2][j+2]!="P") a[i-2][j+2]="B";
        if (i+2 <m && j-2>=0 && a[i+2][j-2]!="P") a[i+2][j-2]="B";
    }
```

Комментарий: и bash-скрипт

1) Каждый тест запускайте отдельно! Если использовать makefile, это не сложно
2) Сначала помечаем клетки под ударом данных башен (ф-ия markB).
3) Затем по очереди в каждую свободную клетку помещаем новую башню и увеличиваем счетчик.