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.

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:

Ввод:	***Вывод:***
	3
FPFP	6
PFPP	
GFGF	
5 3	
FPF	
FFF	
FGG	
PFP	
FPF	
0 0	

****

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

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

70

1	2	3	4	Σ
9	9	11	15	35

заполняется жюри!

5173

ЛИЧНАЯ 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 11.03.17

****

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).

****

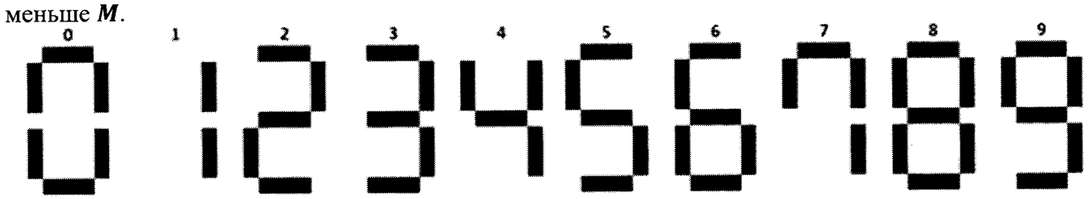
Требования к оформлению задач по программированию:

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

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.

Пример:

Ввод: 86 10

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

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:

```
2
1 1 0 2
1 2 2 3
```

Вывод:

```
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

N2

```
int n,m,k,s=0,i,j,l,c=0,r; cout << "Введите число и ограничение M" << endl;
cin >> n >> m;
r=n;
while(r>0){                    Узнаем длину числа
    r/10;
    s++;
}
int a[s], b[s];
int x[10][10];
for(i=0; i<10; i++){
    for(j=0; j<10; j++){
        x[i][j]=-1
    }
}                              создание таблицы
    x[0][0] = 7;
    x[0][1] = 1;
     . . .
      *
for(i=s-1; i>=0; i--){
    a[i] = n%10;               расписываем
    n = n/10;                  исходное число по цифрам
}
for(i=max(m+1, 10·(s-1)); i<10·(s+1)-1; i++){   l=i;
    for(j=s-1; j>=0; j--){     увеличивая нужное нам число мы его тоже
        b[j] = l%10;           расписываем по
        l = l/10;              цифрам.
    }
    for(j=0; j<s; j++){
        for(k=0; k<10; k++){
            if(x[a[j]][k]==b[j])
                c++;
        }
    }
    if(c==s){
        cout<<i;
        return 0;
    }
}
return 0;
```

* Там, где стоит '*', идут операции типа x[i][j] = 7, чтобы получить таблицу x:

0	7	1	0	-1	-1	-1	-1	-1	-1	-1
1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	3	1	-1	-1	-1	-1	-1	-1	-1	-1
4	4	1	-1	-1	-1	-1	-1	-1	-1	-1
5	5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	6	5	-1	-1	-1	-1	-1	-1	-1	-1
7	7	1	-1	-1	-1	-1	-1	-1	-1	-1
8	8	8	7	6	5	4	3	2	1	0
9	9	7	5	4	3	1	-1	-1	-1	-1

Таблица x показывает, какие цифры могут получиться из цифры i, т.е. в i-ой строке по убыванию стоят цифры, которые могут получиться из i.

если все цифры числа можно получить, то выписываем это число и прерываем программу

9

Чистовик

№4

```
int m=1; n=1; i, j, k;
while (m!=0 && n!=0){ cout<<"Введите M, N"<<endl;
    cin>>m>>n;
    k=0;
    ~~char a[m][n];~~ char a[m][n];   cout<<"заполните таблицу"<<endl;
    for(i=0; i<m; i++){
        for(j=0; j<n; j++){
            cin>>a[i][j];
            if(a[i][j]=='F')
                k++;
        }
    }
    for(i=0; i<m; i++){
        for(j=0; j<n; j++){
            if((a[i][j]=='F') && ((i<m-2 && a[i+1][j]!='P' && a[i+2][j]=='G')
                                  ||(i>1 && a[i-1][j]!='P' && a[i-2][j]=='G')
                                  ||(j<n-2 && a[i][j+1]!='P' && a[i][j+2]=='G')
                                  ||(j>1 && a[i][j-1]!='P' && a[i][j-2]=='G')))
                k--;
            else a[i][j]='G';
        }
    }
    cout<<k<<endl;
}
return 0;
```

Предполагаем, что везде, где есть свободные клетки, можно поставить башню. Дальше проверяем каждую клетку на каждую из 4 сторон. Если где-то нет рядом стены, а дальше идет башня, т.е. свободная клетка под ударом, то кол-во башен уменьшается на 1. В противном случае в свободную клетку ставим башню.

Чистовик

v3

```
#include <math.h>
long long X[200000]
long long n,a,b,c,p,w,i,k,t; cout<<"введите кол-во тестов"<<endl;
cin>>n;
for(i=0; i<n; i++){ cout<<"введите A,B,C,P"<<endl;
   cin>>a>>b>>c>>p;
      t=0; k=0;
      for(w=0; w<[redacted] 2·pow(10,18); w++){
        if((((A·w+b)·w+c) %p==0){
              X[t]=w;
              t++;
              k++;
        }
      }
      [redacted]
      cout<<k<<" "[redacted];
      [redacted];
      for(j=0; j<=t; j++)
         cout<<X[j]<<" ";
}
return 0;
```

Делаем перебор вариантов. Если w удовлетворяет условию $((Aw+B)\cdot w + C)\,\%\,p == 0$, то w заносится в массив.

($((Aw+B)\cdot w + C)$ = t, t = s)

Далее выводим кол-во, затем массив.

~~10~~

11