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

56

1	2	3	4	Σ
6	10	12	∅	28

9350

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 11.3.2017

****

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).

****

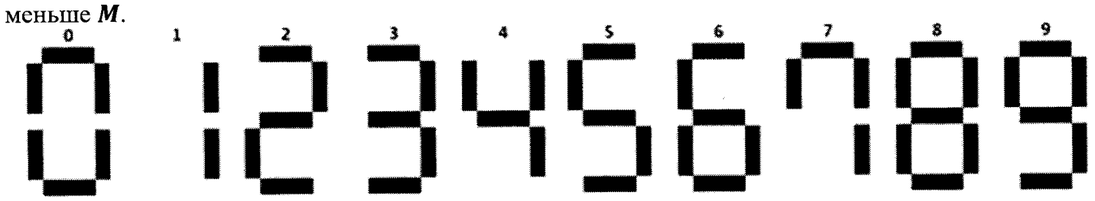
Требования к оформлению задач по программированию:

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

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

Задача №1.

Алгоритм: будем перебирать все возможные слова длины d и проверять в каждом количество неповторяющихся букв.

```
var
  i,d,j : Integer;
  s, ans : String;
  a: Array [0..1000] of Integer;
  word: Array [0..1000] of Integer;
begin
  readln(s);
  readln(s);
  readln(d);
  while d<>0 do
  for i:=0 to (d-1) do
    word[i]:=1;
  while i>-1 do
  begin
    for j:=0 to (s-1) do
      a[j]:=0;
    for j:=0 to (d-1) do
      a[word[j]]+=1;
    flag:=true
    for j:=0 to (s-1) do
      if a[j]>3 then
        flag:=false;
  if flag then
    for j:=0 to (d-1) do
      ans:=ans+s[word[j]];
  writeln(ans);
  ans:='';
  i:=d-1;
  if word[i]<(s+1) then
    word[i]+=1;
  else
  begin
    i:=i-1
    while (i>=0) and (word[i]=s) do
      i:=i-1;
    if word[i]<>-1 then
      for j:=(i+1) to (d-1) do
        word[j]:=1;
  end;
  end;
end.
```

(6)

Задача №3 Числовик

Алгоритм решения:

Площадь фигуры $S = (Aw + B)w = Aw^2 + Bw$

$S + C = Aw^2 + Bw + C \ | \ p$

Заметим, что если такое w существует, то решений у данного выражения бесконечно много.

Пусть выполнено $Aw^2 + Bw + C \ | \ p$, тогда это правда для $(w + p)$

$A(w+p)^2 + B(w+p) + C = Aw^2 + 2Awp + Ap^2 + Bw + Bp + C =$

$= \underbrace{Aw^2 + Bw + C} + p(2Aw + B + Ap) \ | \ p$

$\cdot p$

Рассматриваем $0 \le w < p$

Перебираем все целые значения из данного диапазона и проверяем выполнение условия

```
var
  i, n, a, b, c, p, j, count: Integer;
  S: String
begin
  readln(n);
  for i := 0 to n-1 do                    # цикл по всем наборам
  begin
    read(a, b, c, p);
    S := '';                              # обнуляем счётчик и строку
    count := 0;
    for j := 0 to p-1 do
      if ((a*j*j + b*j + c) mod p = 0) then    # проверка условия
      begin
        count := count + 1
        S := S + ' ' + IntToStr(j);
      end;
    println(count, ' ', S)
  end;
end.
```

Задача 2. Числови

Разберем какво числа ще можем получаваме от другите

0 → 0 ; 1 ; 7
1 → 1
2 → 2
3 → 1, 3
4 → 1, 4
5 → 5
6 → 5, 6
7 → 1, 7
8 → 0, ..., 9
9 → 1, 3, 4, 5, 7, 9

```
var i,n,m
  s_m, s, n

function s_min ( s string) : string
    var
      i: Integer
    begin
      for i:= 0 to len(s)-1 do
        begin
          if s[i] = '3' then
             s[i] := '1';
          if s[i] = '4' then
             s[i] := '1';
          if s[i] = '6' then
             s[i] := '5';
          if s[i] = '7' then
             s[i] := '1';
          if s[i] = '8' then
             s[i] := '0'
          if s[i] = '9' then
             s[i] := '1'
        end;
      s_min := s;
    end;
begin
    read (n);
    read (m);
    sn = str(n);
    sm = str(m);
    if len(sm) < len(sn) then
          sm := s_min (sn);
    else
      begin
```

```
      fl := false;
      i := 0;
      while i < len(sn) do
        begin
          if sn[i] > sm[i] then
          begin
            if (sm[i] <= '6') and (sm[i] >= '3') then
              begin
                if sn[i] = '3' then
                  if (sn[i-1] = '3') or (sn[i-1] = '4')
                or (sn[i-1] = '7') or (sn[i-1] = '9') then
                    sn[i] := '1'
                end;
          else if sm[i] < i then
              if sn[i] = 6 then
                 sn[i] := 5
          end;
          if sn[i] > sm[i] then
             sn := sn[0..i-1] + s_min(sn[i..len(sn)])
             Break;
          if sn[i] < sm[i] then
            begin
              j := i
              sn1 := str(n)
              while (j>=0 and sn[j] = sn1[j]) do
                j += 1.
```

(15)

Продолжение 2

```
  Sn[j] := Sm[j];        # мы нашли наименьший разряд
                         # в котором мы уменьшили число
  Sn := Sn[0..j] + S.min(Sn[j+1]..len(Sn)-1];
 end;
end;          # конец цикла
end;
for i := 0 to len(Sn)-1 do
  print(Sn[i], ' ');
end.
```

комментарий к алгоритму

Спускаемся от старших разрядов к младшим

Если есть возможность уменьшить число в разряде, чтобы оно не превосходило число в след. разряде Ограничения - уменьшаем.

Если в разряде число меньше чем в минимуме - поднимаемся до последнего уменьшенного и увеличиваем его

Если в какой-то момент после уменьшения в разрядах Sn[i] > Sm[i] то конец числа минимизируется.