**Задача №4.**

Володя решил переехать в другой город, но, к сожалению, он не может перевезти с собой все вещи, поэтому оставшиеся ***N*** предметов он решает продать и размещает объявления в интернете. На следующий день он проверил почту и обнаружил, что ему поступило ***M*** предложений о покупке, в каждом из которых покупатель готов приобрести ***K*** предметов, но только в случае, если Володя продаст все из списка сразу. Помогите Володе подсчитать количество предложений, с помощью которых он продаст наибольшее количество вещей.

***Входные данные:***

Первая строка: количество вещей ***N*** и количество предложений о покупке ***M*** через пробел;
Последующие строки: ***M*** предложений, каждое из которых содержит:

- Количество предметов в списке покупателя K,
- Номера вещей из списка продаваемых. Читать, что нумерация начинается с 1.

***Выходные данные:***

Целое число, соответствующее количеству предложений.

**Пример:**

***Ввод:***
4 3
2 1 2
2 2 3
2 3 4

***Вывод:***
2

**********

***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*

---

ГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

6817

| 1 | 2 | 3 | 4 | Σ |
|---|---|---|---|---|
| 6 | 10 | 12 | 18 | 46 |

заполняется жюри!

92

# ПИСЬМЕННАЯ РАБОТА УЧАСТНИКА ОЛИМПИАДЫ ШКОЛЬНИКОВ СПбГУ 2016–2017

заключительный этап

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

Город, в котором проводится Олимпиада Челябинск

Дата 21.02.2017

**********

**Вариант 01**

**********

**Задача №1.**

На планете «Крохотные крошки» в именах жителей не стоят две одинаковые буквы рядом. Необходимо написать оптимальный алгоритм генерации имен жителей планеты, если алфавит планеты состоит из 30 букв. Каждое имя состоит минимум из 3 букв, максимум из 10.

**********

***Требования к оформлению задач по программированию:***

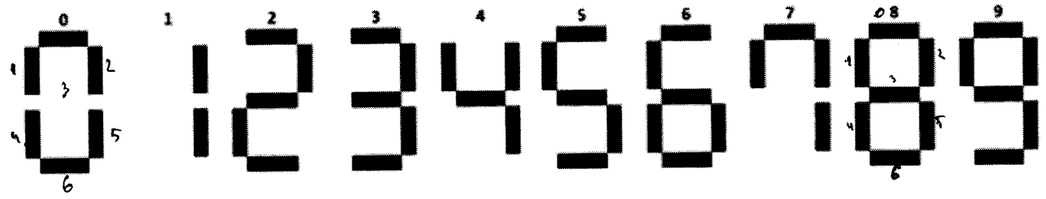
1) *Программы должны быть написаны на одном из языков: C, C++, Pascal*
2) *Полностью оформленная задача должна содержать:*
   - *программу, выполняющую необходимые операции для всех допустимых данных;*
   - *операции с файлами входных и выходных данных **или** понятный пользователю интерфейс ввода исходных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

*Невыполнение вышеуказанных требований влечет за собой снижение получаемых за задачи баллов*

---

## Задача №2.

Цифровое табло, состоящее из семи-сегментных дисплеев, используется для вывода числовых значений. Вы можете ***активировать выключенные*** сегменты дисплея, но не можете ***выключать*** уже активные. Необходимо написать программу или алгоритм на языках C++, Pascal, Basic для определения максимального числа, которое может быть выведено на дисплей и не будет больше 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:*** 25 100

***Вывод:*** 89

**********

***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*

## Задача №3.

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

***Входные данные:***
Первая строка содержит количество тестовых случаев, каждая следующая – тестовый случай с разделенными пробелом значениями $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ и $\boldsymbol{P}$.

***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:***

ширина не может быть равна 0.
2 1
0

**********

***Требования к оформлению задач по программированию:***

3) *Программы должны быть написаны на одном из языков: C, C++, Pascal*
1) *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*

# Продолжение Задачи 4.



```
if(ok)
{ int sum_now=0; // кол-во товаров, которое можно продать сейч.
  for(int i=0;i<indeks.size();i++)
  {
    for(int j=0;j<n;j++)
      sum_now += mass[indeks[i]][j];
    if(sum_now>sum)        // если стоит продать больше, то обновляем ответ;
    { sum=sum_now;
      ans=indeks.size();
    }
  }
}
cout << ans;
return 0;
}
```

8

Задача №4 C++

```
#include <algorithm>
#include <iostream>
#include <vector>
#include <cmath>
using namespace std;
int n, m;
vector<vector<int>> mass;
bool can(int i1, int i2) // проверка, что предложение i1 и i2 не пересекаются.
{
    bool ok = true;
    for(int i=0; i<n; i++)
    {
        if(mass[i1][i]==1 && mass[i2][i]==1)
            ok=false;
    }
    return ok;
}

int main()
{ int sum=0, ans;
 cin >> n >> m;
 mass.resize(m);
 for(int i=0; i<m; i++)
 {
    vector<int> add(n,0);
    int k; // кол-во предметов в i-том предложении.
    cin >> k;
    for(int j=0; j<k; j++)
    { int a;
      cin >> a;
      a--;
      add[a]=1;
    }
    mass[i]=add;
}
// Переберем все возможные варианты выбора предложений и проверим, что выбранные варианты не пересекаются
// Переберем с помощью двоичных масок.
long long to= 1<<(long long)(m); // 2^m
for(long long now=1; now<=to; now++)
{
    vector<int> mask;
    long long sn = now;
    while(sn)          // перевод sn в 2сс
    { mask.push_back(sn%2);
      sn /= 2;
    }
    reverse(mask.begin(), mask.end());
    while(mask.size()<m)
        mask.push_back(0);

    vector<int> indexs; // индексы масок, где стоит 1
    for(int i=0; i<mask.size(); i++)
    { if(mask[i]==1)
        indexs.push_back(i);
    } bool ok=true;
    for(int i=0; i<indexs.size(); i++)
    {
        for(int j=i+1; j<indexs.size(); j++)
        { if(!can(indexs[i], indexs[j]))
            ok=false;
        }
    }
```

Задача №3 C++

```
#include <iostream>
#include <cmath>
#include <algorithm>
using namespace std;
int main()
{
  int q; // кол-во запросов
  cin >> q;
  for(int i=0; i<q; i++)
  {
    int a, b, c, p;
    cin >> a >> b >> c >> p;
// Пусть x - длина квадрата, y - ширина, (x>0, y>0, x,y - целые), тогда по условию задачи
// y = ax+b (прямая)
// (ax^2+bx+c) = p * k, k ∈ Z. (парабола).
// Нам подходят такие x, которые принадлежат обоим функциям (их не больше 2х)
// Найдем точки пересечения. Двух функций и проверим, что y (целый) координаты кратна p.
// Ax + B = Ax^2 + Bx + C
// Ax^2 + x(B-A) + (C-B) = 0   Целое квадратное ур-ие.
//  A1     B1      C1

  int A1 = A,
      B1 = B - A,
      C1 = C - B;

  double D = B1 * B1 - 4 * A1 * C1;
  if (D < 0)
  { cout << 0 << "\n";
    continue;
  }

  double x1 = (-B1 + sqrt(D)) / (double)(2 * A1);
  double x2 = (-B1 - sqrt(D)) / (double)(2 * A1);
  double y1 = a * x1 + b;
  double y2 = a * x2 + b;
  bool ok1 = true,
       ok2 = true;
  if (x1 <= 0 || y1 <= 0 || (int)(x1) != x1 || (int)(y1) % p != 0)
      ok1 = false;
  if (x2 <= 0 || y2 <= 0 || (int)(x2) != x2 || (int)(y2) % p != 0)
      ok2 = false;
                    // выводим ответ
  if(ok1 && ok2)
  {
    if(x1 == x2) // если точка касания - вершина параболы, но точки совпадут.
      cout << 1 << " " << x1 << "\n";
    else
      cout << 2 << " " << (int)min(x1,x2) << " " << (int)max(x1,x2) << "\n";
  }
  else if(ok1)
    cout << 1 << " " << x1 << "\n";
  else if(ok2)
    cout << 1 << " " << x2 << "\n";
  else cout << 0 << "\n";
}
return 0;
}
```

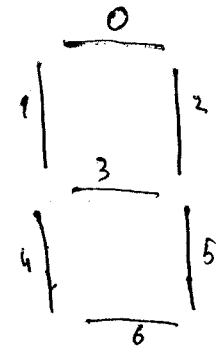
(12)

проверка на целость

проверка на кратность p.

// подставляем и находим y координаты.

# Задача №2. C++




cif - массив, в котором для каждой цифры отмечено какие сегменты должны быть горящими.

```
#include <iostream>
#include <cmath>
#include<string>
#include<vector>
using namespace std;
vector<vector<int>> cif= { { 1, 1, 1, 0, 1, 1, 1 }, // 0
                           { 0, 0, 1, 0, 0, 1, 0 }, // 1
                           { 1, 0, 1, 1, 1, 0, 1 }, // 2
                           { 1, 0, 1, 1, 0, 1, 1 }, // 3
                           { 0, 1, 1, 1, 0, 1, 0 }, // 4
                           { 1, 1, 0, 1, 0, 1, 1 }, // 5
                           { 1, 1, 0, 1, 1, 1, 1 }, // 6
                           { 1, 1, 1, 1, 0, 1, 0 }, // 7
                           { 1, 1, 1, 1, 1, 1, 1 }, // 8
                           { 1, 1, 1, 1, 0, 1, 1 } }; // 9
//можем ли из горящего а сделать b
bool can(char a, char b)
{ //перевод символа в цифру.
  int aa=a-'0', bb=b-'0';
  bool ok=true;
  for(int i=0;i<7;i++)
  {
   if(cif[aa][i]==1 &&
      cif[bb][i]==0)
     ok=false;
  }
  return ok;
}

int main()
{  string ans; // строка с ответом.
   string a, m; // т.к. не даны ограничения на a и m, то лучше работать со строками,
                // чтобы избежать переполнения.
   cin>>a>>m;
   bool more=false; // длина m > длины a.
   if(m.size() > a.size())
      more=true;
   for(int i=0; i< a.size(); i++) // последовательно проходим и собираем ответ.
   {
    char now; // перебираем ~~~~~~~~~~ цифры в порядке убывания и пробуем подставить.
    if(more)
      now='9';
    else now=m[i];
    while(now >='0')
    { if( can(a[i],now))      // если получилось зажечь сегмента как, чтобы горела цифра now, но
      { ans += max(now, a[i]); // добавляем ее в ответ, если она больше того, что уже стоит.
        break;
      }
      now--;
    }
   }
   if(ans.size() != a.size())
     cout << "невозможно составить";
   else
     cout << ans;
   return 0;
}
```

(10)

Задача 1. C++

```
#include <vector>
#include <iostream>

using namespace std;

vector<vector<int>> a; // массив, в который будут записываться последовательности индексов дуг,
                       // которые нам подходят в индексации с 0!
vector<int> now; // текущий массив из индексов дуг, меняется во время рекурсии.
void rec()
{
  if(now.size() > 10) // если слово уже слишком большое, то заканчиваем эту ветвь рекурсии.
    return;
  if(now.size() >= 3) // нашли подходящую строку, добавим ее.
    a.push_back(now)

  for(int next_char=0; next_char<30; next_char++)
  {
    if(now.back() != next_char) // если добавляемый символ не равен последнему, то мы можем его добавить и запустить новую ветвь рекурсии.
    {
      now.push_back(next_char);
      rec();
      now.pop_back();
    }
  }
}

int main()
{
  rec(); // Запускаем рекурсию
  // Выводим все найденные слова
  for(int i=0; i<a.size(); i++)
  {
    for(int j=0; j<a[i].size(); j++)
      cout << a[i][j] << " ";
    cout << "\n";
  }
  return 0;
}
```

(6)