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

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    int n; cin>>n; int m; cin>>m; vec.resize(m+1)
    for(int i=0; i<m; i++)
        int k;
        cin>>k;
        for(int i=0; i<k; i++) int a; cin>>a; vec[i].push_back(a)
vector<int, bool> common(n+1);
vector<
```

vector<vector<int>> vec;

1) считываем все данные
2) dp[0] = {0, 0}
   dp[i] = {i, vec[i].size()}
   - количество данных

«объявление массив, где хранится информация о купивших товарах

см. на доп. листе :)

ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

1	2	3	4	Σ
4	9	3	15	31

2

8802

заполняется жюри!

62

ПИСЬМЕННАЯ РАБОТА УЧАСТНИКА ОЛИМПИАДЫ ШКОЛЬНИКОВ СПбГУ 2016–2017

заключительный этап

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

Город, в котором проводится Олимпиада Киров

Дата 03.03.2017

****

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

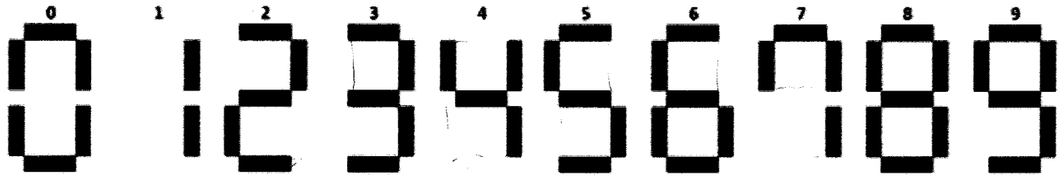
```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    freopen("input.txt", "r", stdin); // открываем входные данные
    freopen("output.txt", "w", stdout); // выходные данные output.txt
    string alf;
    cin >> alf; // читаем алфавит
    int count_str = 3 + mt_rand % 10; // случайное число от 3 до 10 — длина имени
    int str = ""; // строка, где будет храниться имя
    str += alf[0 + mt_rand() % 30]; // берем случайную букву в алфавите от 1 до 30
    for(int i=1; i<count_str; i++)
        int a = alf[0 + mt_rand % 30]; // выбираем случайную букву
        int b = alf[0 + mt_rand % 30]; // выбираем случайную букву выбираем новую.
        if(a==b) b = (char)(int)b++). // если они равны, то в следующий раз буква
        if(str[i-1] != a) // если последние буквы в текущем не равны a, то берем a
            str += a
        else
            str += b // иначе берем b (a ≠ b по проверке)
cout << "Имя:" << str; // выводим имя.
```

4

вероятность выбора // одной и той же a и b в имени очень мало.

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

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    freopen("input.txt", "r", stdin);  // входные данные
    freopen("output.txt", "w", stdout); // выходные данные
    map<char, vector<char>>; //запишем всевозможные цифры, которые можем получить
    map['0'] = {'0', '8'}; map['1'] = {'1', '7'}; map['2'] = {'2', '8'}; map['3'] = {'3', '8', '9'};
    map['4'] = {'4', '8', '9'}; map['5'] = {'5', '8', '9'}; map['6'] = {'6', '8'}; map['7'] = {'7', '8', '9'}; map['8'] = {'8'}
    map['9'] = {'8', '9'};
    string n; int m; cin >> n >> m;
    for(int i=0; i<n.size(); i++)
    {
        for(auto q: mp[n[i]]) // идем по возможным цифрам для i цифры
        {
            string new_string = n; //изначально n
            new_string[i] = q;
            int num = 0;
            for(int i=0; i<new_string.size(); i++)
                num += pow(10, new_string.size()-i-1) * (new_string[i] - '0'); // преобразовываем строку в число
            if(num <= m)
                max_num = max(num, max_num);
        }
    }
    string n1 = "";
    while(max_num != 0)
        n1 = (char)(max_num % 10) + '0') + n1; // преобразуем макс. число в строку
        max_num /= 10;
    if(n1 != "")
        n = n1; // присвоим получившееся максимум строкой строке
}
cout << n; // вывод наш значения
```

1) Запишем все возможные цифры, которые можно получить из цифр от 0 до 9
2) пройдем по каждому числу. будем перебирать для i-цифры. заменим на очередную возможную цифру, преобразуем в число и сравним с M. Если подходит то выведем из этих чисел максимум

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 $\boldsymbol{A, B, C}$ 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:	***Вывод:***
2 1 1 0 2 1 2 2 3	2 0 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

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    int n; //кол-во тестов
    long long A, B, C, P;
    cin >> n;
    for(int i=0; i<n; i++)
    {
        cin >> A >> B >> C >> P;
        bool flag = false;
        int count = 0; vector<int> ans;
        for(int j=1; j<P; j++) // считываем данные
            if((A*j + B) * j + C) % P == 0) // проверяем условие
            {
                count++; ans.push_back(j);
                flag = true
            }
        if(!flag) cout << "0";
        else { cout << count << endl; for(auto q: ans) cout << q << " "; }
    }
}
```

задача № 4

```
#include <bits/stdc++.h>
using namespace std;
int main()
{   freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
    int n, m;
    cin>>n>>m;
    vector<vector<int>> vec(m+1);
    for(int i=0; i<m; i++)
    {   int k;
        cin>>k;
        for(int j=0; j<k; j++)
        {   int a;
            cin>>a;
            vec[i].push_back(a);
        }
    }
    vector<pair<bool,int>> common;
    common.resize(n+1);
    vector<pair<int,int>> dp(n+1);
    dp[0] = {0, 0};
    dp[1] = {1, vec[0].size()};
    for(int i=1; i<n; i++)
    {   vector<pair<bool,int>> com = common;
        for(auto q : vec[i])
        {   com[q-1].first = true;
            com[q-1].second = i;
        }
        map<int,int>mp; int count=0
        for(auto q : com)
        {   mp[q.second]=1;
            count+=q.first;
        }
        if(dp[i-1].second < count)
        {   dp[i].second = count;
            dp[i].first = mp.size();
            common = com;
        }
        else
            dp[i]=dp[i-1];
    }
    cout<< dp[n-1].first;
}
```

1) `if(com[q-1].first==false) com[q-1].first=true; com[q-1].second=i;`

`else { int a_i = com[q-1].second; for(auto w : com) if(w.second==a_i) { w.first=false; w.second=0; } com[q-1].first=true; com[q-1].second=i; }`

// считываем данные

// объявляем массив, где хранится инф-ия о заказах предметах и их "владельцах". Все инф. с 0

// dp - массив, где будет храниться рез-тат, dp[i].first - максимальное кол-во заказов; dp[i].second - мин. кол-во товаров, которые можно продать

// исходные данные

// объявляем локальный common.

// перезаписываем com новыми заказами

// считаем кол-во товаров, которые можно продать и кол-во заказов в целом

// улучшаем результат

// оставляем все, как было на прошлом шаге

Алгоритм. Метод динамического программирования

1) считываем
2) создаем common, где хранится вся инфор. про товары и dp, где хранится ответ
3) идет по каждому предложению о покупке, если никакие товары не пересекаются с common отмечаем эти товары
   иначе перезаписываем.
4) смотрим, не ухудшился ли этот запуск как ответ.
5) выводим ответ.

(15)