Задачи финала - условия

A. Нормализация пути

ограничение по времени на тест
2 секунды
ограничение по памяти на тест
64 мегабайта
ввод
стандартный ввод
вывод
стандартный вывод

Макс изучает файловую систему Ubuntu. Он уже знаком со следующими её особенностями:

- Файловая система представляет собой иерархию каталогов (папок);
- Имя каталога является непустой строкой, состоящей из латинских букв и знаков препинания. Буквы различного регистра считаются разными;
- Корневой каталог обозначается как /;
- От корневого каталога до любого другого каталога существует уникальный путь. В записи пути каталоги разделяются знаком /. Так, если каталог bar находится в каталоге foo, а каталог foo — в корневом каталоге, то полный путь до каталога bar выглядит следующим образом: /foo/bar/;
- Запись ./ обозначает текущий каталог;
- Запись ../ обозначает родительский каталог, а для корневого каталога — текущий каталог.

Макс долго перемещался по файловой системе в поисках нужного ему каталога, и в конце концов нашёл его. Теперь Макс хочет сохранить путь до каталога на будущее, но предварительно нужно его нормализовать, то есть записать в кратчайшем возможном виде.

Помогите Максу выполнить нормализацию пути.

Входные данные

Ввод содержит строку S ($1 \le |S| \le 10^5$), состоящую из латинских букв, знаков препинания и символов / — корректный путь до некоторого каталога.

Выходные данные

Выведите одну строку — эквивалентный путь, содержащий минимально возможное количество символов.

Примеры
Входные данные
```
/foo/bar/../baz/
```
Выходные данные

```
/foo/baz/
```
Входные данные
```
/foo/bar/./baz/
```
Выходные данные
```
/foo/bar/baz/
```

B. На соревнования — на такси. Снова

ограничение по времени на тест
8 секунд
ограничение по памяти на тест
64 мегабайта
ввод
стандартный ввод
вывод
стандартный вывод

Как вы помните, Макс и другие студенты, находясь в командировке, часто пользуются услугами такси. На этот раз ребята отправились в другой соседний регион, где проводится знаменитый Апрельский Чемпионат.

Университет-организатор Апрельского Чемпионата имеет N корпусов, пронумерованных от 1 до N. Корпуса соединены сетью из M дорог, проезд по i-й из которых занимает T_i минут.

Макс и его друзья поселились в гостинице, находящейся в непосредственной близости от корпуса, имеющего номер S. К сожалению, никто из ребят не запомнил, в каком именно корпусе будет проходить чемпионат, поэтому сейчас они изучают карту города и планируют, как лучше всего добраться до того или иного корпуса.

Оказалось, что в городе, в котором проводится Апрельский Чемпионат, службы такси имеют ряд особенностей:

- Во-первых, этих служб очень много (можно считать, что сколь угодно большая группа пассажиров всегда сможет заказать такси).
- Во-вторых, несмотря на большое количество служб, в каждой из них можно заказать только одну машину.
- В-третьих, водители разных служб такси не очень жалуют друг друга, и поэтому обязательно поедут к месту назначения различными путями (пути A и B считаются различными, если существует хотя бы одна дорога, входящая в путь A и не входящая в путь B или наоборот).

Макс и другие студенты долго чесали в затылках, силясь понять бесконечную мудрость местной сферы транспортного обслуживания.

Сейчас ребята хотят выяснить, сколько существует корпусов, до которых они могут добраться одновременно и за минимально возможное время, если воспользуются местными службами такси и начнут поездку одновременно. Помогите им найти ответ на этот вопрос. Помните, что во всех такси 4 места для пассажиров.

Входные данные

Первая строка содержит целые числа N и M ($1 \le N, M \le 10^5$, $0 \le M \le 10^5$) — соответственно количество корпусов университета и соединяющих их дорог.

Следующие M строк описывают дороги. Каждая из них содержит целые числа A_i, B_i и T_i ($1 \le A_i, B_i \le N$, $1 \le T_i \le 10^5$) — соответственно номера корпусов, которые соединяет дорога, и время проезда по ней в минутах.

Следующая строка содержит целые числа S и K ($1 \le S \le N$, $1 \le K \le 1000$) — соответственно номер корпуса, рядом с которым расположена гостиница, и количество студентов.

Выходные данные

Выведите одно или более целых чисел — номера корпусов, до которых все студенты, стартовав одновременно, могут доехать также одновременно и за минимально возможное время. Номера следует выводить в порядке возрастания.

Примеры
Входные данные
```
4 5
1 2 1
1 3 1
2 4 1
3 4 1
2 3 2
1 8
```
Выходные данные
```
1 4
```
Входные данные
```
5 7
1 2 1
1 3 2
1 4 3
1 5 4
2 3 1
3 4 1
4 5 1
1 10
```
Выходные данные
```
1 4 5
```

C. Radar Defence

ограничение по времени на тест
2 секунды
ограничение по памяти на тест
64 мегабайта
ввод
стандартный ввод
вывод
стандартный вывод

Вы играете в игру «Radar Defence» — новейший продукт компании MAXSOFT.

В точке (0; 0) координатной плоскости расположена радарная башня, которую вам необходимо защищать. В момент времени 0 появляются N вражеских ракет, i-я из которых

имеет начальные координаты (X_i; Y_i). Каждая ракета движется по прямой к радарной башне и пролетает единичный отрезок координатной сетки за 1 секунду.

Радарная башня оснащена лазерной установкой, вращающейся против часовой стрелки. В момент времени 0 установка направлена вдоль оси *Ox*. За первую секунду установка может уничтожить любую одну ракету, расположенную в I координатной четверти. За вторую секунду установка может уничтожить любую одну ракету, расположенную во II координатной четверти. За третью секунду установка может уничтожить любую одну ракету, расположенную в III координатной четверти. За четвёртую секунду установка может уничтожить любую одну ракету, расположенную в IV координатной четверти. За пятую секунду установка вновь может уничтожить любую одну ракету, расположенную в I координатной четверти, и так далее.

Попробуйте определить, сможет ли лазерная установка уничтожить все ракеты, прежде чем они долетят до радара.

Входные данные

Первая строка содержит целое число N ($1 \le N \le 10^5$) — количество ракет.

Следующие N строк описывают ракеты. Каждая из них содержит целые числа X_i и Y_i ($-10^9 \le X_i, Y_i \le 10^9$) — начальные координаты ракеты.

Начальные координаты ракет не принадлежат координатным осям.

Выходные данные

Выведите YES, если лазерная установка уничтожит все ракеты. В противном случае выведите NO.

Примеры
Входные данные
```
4
5 7
3 1
-1 2
-8 4
```
Выходные данные
```
YES
```
Входные данные
```
4
2 1
3 2
2 -2
-3 3
```
Выходные данные
```
NO
```

D. Amuz Deluxe

ограничение по времени на тест
2 секунды
ограничение по памяти на тест

64 мегабайта
ввод
стандартный ввод
вывод
стандартный вывод

Компания MAXSOFT создаёт и казуальные игры. Наиболее популярной из них является «Amuz Deluxe».

В этой игре игрок управляет пушкой, стреляющей цветными шариками. Перед пушкой медленно движется змейка из шариков; шарик, выстреленный из пушки, занимает место между некоторыми двумя шариками в змейке.

Если после выстрела в змейке оказались три шарика одного цвета, расположенных подряд, то они уничтожаются. Передняя часть змейки при этом откатывается назад, к хвосту, после чего снова может получиться смежная группа из трёх или более шариков одного цвета, которая также уничтожается, и так далее.

Вам дано описание змейки сразу после выстрела игрока. Определите, сколько шариков будет уничтожено. Гарантируется, что в змейке имеется не более одной группы из трёх смежных шариков одинакового цвета.

Входные данные

В первой строке содержится целое число N ($3 \le N \le 10^5$) — количество шариков в змейке.

Вторая строка содержит N целых чисел C_1, C_2, ... C_N ($1 \le C_i \le 100$) — цвета шариков.

Выходные данные

Выведите одно целое число — количество уничтоженных шариков.

Примеры
Входные данные
```
10
1 2 1 3 3 3 1 2 2 1
```
Выходные данные
```
3
```
Входные данные
```
10
2 2 1 3 3 3 1 1 2 1
```
Выходные данные
```
9
```

E. Собеседование

ограничение по времени на тест
2 секунды
ограничение по памяти на тест
64 мегабайта
ввод
стандартный ввод
вывод

стандартный вывод
Макс был недоволен задачами прошлого контеста, но как знать — может, эта задача ему всё-таки понравится?

Макс хочет устроиться в Очень Серьёзную Контору на Очень Серьёзную Должность. Разумеется, для этого ему нужно пройти достаточно сложное собеседование. Одним из важных умений, которые Макс должен продемонстрировать, является способность быстро считать в уме.

На столе перед Максом лежат (N - 1) карточек, на которых записаны все целые числа от 2 до N. Интервьюер задаёт Максу M вопросов.

В каждом вопросе интервьюер выбирает K различных карточек, а Макс должен быстро сообразить, с какой степенью произведение чисел на выбранных карточках входит в произведение чисел на всех карточках.

Помогите Максу пройти собеседование!

Входные данные

Первая строка содержит целые числа N, M и K ($1 \le N \le 10^8$, $1 \le M \le 50$, $1 \le K \le min(N, 50)$) — соответственно максимальное число, написанное на карточке, количество вопросов интервьюера и количество карточек, которые выбирает интервьюер в каждом вопросе.

Следующие M строк описывают вопросы интервьюера. Каждая из них содержит K различных целых чисел A_i ($2 \le A_i \le N$) — числа на выбранных карточках.

Выходные данные

Выведите M целых чисел — максимальные степени, с которыми произведения чисел на карточках, выбранных в каждом вопросе, входят в произведение чисел на всех карточках.

Примеры
Входные данные
```
10 3 3
2 3 5
2 5 7
2 4 8
```
Выходные данные
```
2
1
1
```
Входные данные
```
100 3 3
2 3 5
2 7 11
2 4 8
```
Выходные данные
```
24
9
16
```

F. Шоколадка

ограничение по времени на тест
2 секунды
ограничение по памяти на тест
64 мегабайта
ввод
стандартный ввод
вывод
стандартный вывод

У Макса есть шоколадка размером $H \times W$, состоящая из маленьких долек размером 1×1. Макс хочет угостить шоколадкой Владимира, а для этого ему нужно разломить шоколадку на две как можно более равные части.

Макс может разламывать шоколадку по вертикали или горизонтали, но линия разлома не может проходить сквозь дольки. Помогите Максу посчитать, на сколько долек больше будет в одной из двух частей шоколадки, если Макс разделит её оптимально.

Входные данные

Ввод содержит целые числа H и W ($1 \le H, W \le 100$, $H \cdot W > 1$) — размеры шоколадки.

Выходные данные

Выведите одно целое число — минимально возможную разность количества долек в двух частях, на которые можно разломить шоколадку.

Примеры
Входные данные
```
6 4
```
Выходные данные
```
0
```
Входные данные
```
3 5
```
Выходные данные
```
3
```

G. Супермаркет

ограничение по времени на тест
5 секунд
ограничение по памяти на тест
64 мегабайта
ввод
стандартный ввод
вывод
стандартный вывод

После долгих попыток устроиться на работу программистом, Макс наконец получил должность администратора в супермаркете. Заметив, что клиенты магазина часто подолгу

ожидают своей очереди при оплате покупок, Макс решил создать автоматизированную систему, распределяющую покупателей по кассам.

В супермаркете имеются N касс, оператор i-й из которых пробивает одну покупку за время T_i. К кассам последовательно подходят M покупателей, у j-го из них в корзине находятся A_j покупок.

Каждого покупателя нужно направить к той кассе, которая начнёт его обслуживать раньше всех остальных. Если подходящих касс несколько, выбирается касса с наименьшим номером.

Напишите для Макса программу, которая подскажет каждому из покупателей, какую кассу ему следует выбрать.

Входные данные

Первая строка содержит целое число N ($1 \le N \le 10^5$) — количество касс.

Вторая строка содержит N целых чисел T_i ($1 \le T_i \le 10^5$) — время, за которое операторы каждой из касс пробивают одну покупку.

Третья строка содержит целое число M ($1 \le M \le 10^5$) — количество покупателей.

Четвёртая строка содержит M целых чисел A_i ($0 \le A_i \le 10^5$) — количество покупок у каждого из покупателей.

Кассы нумеруются от 1 до N в порядке описания во входных данных.

Выходные данные

Выведите M целых чисел — номера касс, которые должны обслуживать каждого из покупателей.

Примеры

Входные данные

```
3
2 3 2
6
2 3 2 3 2 3
```

Выходные данные

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

Входные данные

```
5
2 3 2 3 1
7
2 3 5 2 5 3 3
```

Выходные данные

```
1 2 3 4 5 1 5
```

H. Разрядка и трансляции

ограничение по времени на тест
10 секунд
ограничение по памяти на тест
64 мегабайта
ввод
стандартный ввод
вывод
стандартный вывод

Сегодня большой день для любителей онлайн-игр — всё мировое сообщество следит за финальными матчами чемпионата по знаменитой многопользовательской игре. У Макса, разумеется, всё не как у людей — в этот знаменательный день он умудрился оказаться в поезде, да ещё и с почти разряженным ноутбуком!

Разумеется, Макс заранее знал о чемпионате и составил список из N матчей, которые он хотел бы посмотреть. Матчи чемпионата транслируются в Интернете: i-й матч начинается в момент времени L_i и заканчивается в момент времени R_i. Несколько матчей могут транслироваться одновременно.

К сожалению, ноутбук Макса вот-вот разрядится: заряда аккумулятора хватит на непрерывную работу в течение M единиц времени. Макс пока выключил свой ноутбук: он решил, что включит его в определённый момент времени и будет смотреть матчи, не отключая ноутбук, до тех пор, пока он не разрядится.

Подскажите Максу, когда именно включить ноутбук, чтобы он сумел посмотреть как можно больше матчей целиком. При необходимости Макс может смотреть несколько трансляций одновременно — на скорость разрядки ноутбука это не влияет.

Входные данные

Первая строка содержит целое число N ($1 \le N \le 10^5$) — количество матчей, трансляцию которых Макс хотел бы посмотреть.

Следующие N строк описывают матчи. Каждая из них содержит целые числа L_i и R_i ($1 \le L_i \le R_i \le 10^9$) — соответственно момент начала и момент окончания матча.

Следующая строка содержит целое число M ($1 \le M \le 10^9$) — количество единиц времени, в течение которых ноутбук Макса может работать, прежде чем разрядится.

Выходные данные

Выведите одно целое число — максимальное количество матчей, которые Макс сможет посмотреть целиком.

Примеры
Входные данные

```
4
0 60
80 100
50 75
```

```
95 120
80
```
Выходные данные
```
3
```
Входные данные
```
4
0 60
80 100
50 75
95 120
40
```
Выходные данные
```
2
```

Решения задач участника Корнеев Андрей Викторович (1 место)

A. Нормализация пути

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <queue>
#include <string>
#include <string.h>
#include <stack>

using namespace std;
const int N = 1e5 + 100;
#define lli long long int
#define MP make_pair

char s[N];

int main() {
#ifdef FILE_IO
  freopen("input.txt", "r", stdin);
  freopen("output.txt", "w", stdout);
#endif
  int n;
  cin >> n;
  stack<char*> st;
  gets(s);
  int start = -1;

  for (int i = 0; s[i]; ++i) {
    if (s[i] == '/') {
      s[i] = 0;
      if (start != -1) {
        if (start + 1 == i && s[start] == '.') {
        } else if  (start + 2 == i && s[start] == '.' && s[start+1] == '.') {
          if (st.size()) st.pop();
        }
        else {
          st.push(s + start);
```

```
        }
      }
      start = -1;
    } else
    if (start == -1) start = i;
  }
  cout << '/';

  vector<char*> v;
  while(st.size()) { v.push_back(st.top()); st.pop(); }
  reverse(v.begin(), v.end());
  for (int i = 0; i < v.size(); ++i) {
    printf("%s/", v[i]);
  }

  return 0;
}
```

C. Radar Defence

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <queue>
#include <string>
#include <string.h>
#include <stack>

using namespace std;
const int N = 1e5 + 100;
#define lli long long int
#define MP make_pair

vector<pair<int, int> > v[2][2];

bool cmp(pair<int, int> &a, pair<int, int> &b) {
  lli d1 = a.first * 1ll * a.first + a.second * 1ll * a.second;
  lli d2 = b.first * 1ll * b.first + b.second * 1ll * b.second;
  return d1 < d2;
}

int main() {
#ifdef FILE_IO
  freopen("input.txt", "r", stdin);
  freopen("output.txt", "w", stdout);
#endif
  int n;
  cin >> n;
  for (int i = 0; i < n; ++i) {
    int x, y;
    cin >> x >> y;
    v[x > 0][y > 0].push_back(MP(x, y));
  }

  for (int dx = 0; dx < 2; ++dx) for (int dy = 0; dy < 2; ++dy) {
    sort(v[dx][dy].begin(), v[dx][dy].end(), cmp);
    int sTime = 0;
    if (dx + dy == 0) sTime = 2;
    else if (dx && !dy) sTime = 3;
    else if (!dx && dy) sTime = 1;
```

```
    for (int i = 0; i < v[dx][dy].size(); ++i, sTime += 4) {
      lli d =  v[dx][dy][i].first * 1ll * v[dx][dy][i].first + v[dx][dy][i].s
econd * 1ll * v[dx][dy][i].second;
      if (d < sTime * 1ll * sTime) {
        cout << "NO";
        return 0;
      }
    }
  }
  cout << "YES";
  return 0;
}
```

D. Amuz Deluxe

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <queue>
#include <string>
#include <string.h>
#include <stack>

using namespace std;
const int N = 1e5 + 100;
#define lli long long int
#define MP make_pair

vector<pair<int, int> > v[2][2];

bool cmp(pair<int, int> &a, pair<int, int> &b) {
  lli d1 = a.first * 1ll * a.first + a.second * 1ll * a.second;
  lli d2 = b.first * 1ll * b.first + b.second * 1ll * b.second;
  return d1 < d2;
}

int main() {
#ifdef FILE_IO
  freopen("input.txt", "r", stdin);
  freopen("output.txt", "w", stdout);
#endif
  int n;
  cin >> n;
  vector<int> v(n);
  for (int i = 0; i < n; ++i) cin >> v[i];
  int st = 0;
  for (st = 0; st < n; ++st) {
    if (v[st] == v[st + 1] && v[st] == v[st + 2]) break;
  }
  int l = st, r = st + 1;
  while (l >= 0 && r < n) {
    int i = l, j = r;
    if (v[i] != v[j]) break;
    int color = v[i];
    int cnt = 0;
    while (i >= 0 && v[i] == color) {
      --i; ++cnt;
    }
    while (j < n && v[j] == color) {
      ++j; ++cnt;
```

```
    }
    if (cnt >= 3) {
      l = i; r = j;
    } else break;
  }
  cout << r - l - 1;
  return 0;
}
```

E. Собеседование

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <queue>
#include <string>
#include <string.h>
#include <stack>
#include <queue>
#include <functional>

using namespace std;
const int N = 1e5 + 100;
#define lli long long int
#define MP make_pair



int main() {
#ifdef FILE_IO
  freopen("input.txt", "r", stdin);
  freopen("output.txt", "w", stdout);
#endif
  int n, m, k;
  cin >> n >> m >> k;
  for (int i = 0; i < m; ++i) {
    map<int, int> v;
    for (int j = 0; j < k; ++j) {
      int t;
      cin >> t;
      for (int d = 2; d *d <= t; ++d) {
        while(t % d == 0) {
          v[d]++;
          t /= d;
        }
      }
      if (t > 1) v[t]++;
    }
    lli ga = -1;
    for (map<int, int>::iterator it = v.begin(); it != v.end(); ++it) {
      lli ans = 0;
      int p = 1;
      for (lli test = it->first ; test <= n; test = test * it->first, ++p) {
        //if (p >= it->second)
        ans += n / test;
      }
      ans /= it->second;
      if (ga == -1) ga = ans;
      ga = min(ga, ans);
    }
```

```
    cout << ga << endl;
  }

  return 0;
}
```

F. Шоколадка

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <queue>
#include <string>
#include <string.h>
#include <stack>

using namespace std;
const int N = 1e5 + 100;
#define lli long long int
#define MP make_pair

int a(int n, int m) {
  int l = (n / 2) * m;
  int r = (n + 1) / 2 * m;
  return r - l;
}

int main() {
#ifdef FILE_IO
  freopen("input.txt", "r", stdin);
  freopen("output.txt", "w", stdout);
#endif
  int n, m;
  cin >> n >> m;

  cout << min(a(n, m), a(m, n));
  return 0;
}
```

G. Супермаркет

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <queue>
#include <string>
#include <string.h>
#include <stack>
#include <queue>
#include <functional>

using namespace std;
const int N = 1e5 + 100;
#define lli long long int
#define MP make_pair
```

```
int main() {
#ifdef FILE_IO
  freopen("input.txt", "r", stdin);
  freopen("output.txt", "w", stdout);
#endif
  int n;
  cin >> n;
  vector<lli> t(n);
  priority_queue<pair<lli, lli>, vector<pair<lli, lli> >, greater<pair<lli, l
li> > > q;
  for (int i = 0; i < n; ++i) {
    cin >> t[i];
    q.push(MP(0, i));
  }
  int m;
  cin >> m;
  for (int i = 0; i < m; ++i) {
    lli cnt;
    cin >> cnt;
    pair<lli, lli> info = q.top(); q.pop();
    cout << info.second+1 << ' ';
    info.first += t[info.second] * cnt;
    q.push(info);
  }
  return 0;
}
```

H. Разрядка и трансляции

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <queue>
#include <string>
#include <string.h>
#include <stack>
#include <queue>
#include <functional>

using namespace std;
const int N = 1e5 + 100;
#define lli long long int
#define MP make_pair



int main() {
#ifdef FILE_IO
  freopen("input.txt", "r", stdin);
  freopen("output.txt", "w", stdout);
#endif
  int n;
  cin >> n;
  vector<pair<int, int> > v(n);
  for (int i = 0; i < n; ++i) cin >> v[i].second >> v[i].first;
  sort(v.begin(), v.end());
  lli m;
  cin >> m;
  int cnt = 0, ans = 0;
  priority_queue<int, vector<int>, greater<int> > q;
```

```
  for (int i = 0; i < n; ++i) {
    ++cnt;
    q.push(v[i].second);
    while(q.size() && (v[i].first - q.top()) > m) {
      q.pop();
      --cnt;
    }
    ans = max(ans, cnt);
  }
  cout << ans;
  return 0;
}
```

Решения задач участника Горшков Даниил Александрович (2 место)

A. Нормализация пути

```
#include <iostream>
#include <stdlib.h>
#include <set>
#include <map>
#include <vector>
#include <cctype>
#include <queue>
#include <math.h>
#include <string>
#include <string.h>
#include <algorithm>
#include <stdio.h>
#include <deque>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef double db;
#define mp make_pair
#define rt return
#define rt0 return 0
#define rt1 return 1
const ll inf = 1000000007;
const int sz = 100100;

int h, w;
string s;
vector<string> v, res;

int main() {
#ifdef _DEBUG
  freopen("in.txt", "r", stdin);   freopen("out.txt", "w", stdout);
#endif

  cin >> s;
  if(s.back() != '/')
    s.push_back('/');
  int l = 1, r = 1;
  for(int i = 1; i < s.size(); ++i){
    if(s[i] != '/')
      r = i;
    else {
```

```
      v.push_back(s.substr(l, r+1 - l));
      l = i + 1;
      r = l;
    }
  }
  for(int i = 0; i < v.size(); ++i){
    if(v[i] == "..") {
      if(!res.empty())
        res.pop_back();
    } else if(v[i] == ".") {
      continue;
    } else
      res.push_back(v[i]);
  }
  cout << "/";
  for(int i = 0; i < res.size(); ++i){
    cout << res[i] << "/";
  }
  return 0;
}
```

C. Radar Defence

```
#include <iostream>
#include <stdlib.h>
#include <set>
#include <map>
#include <vector>
#include <cctype>
#include <queue>
#include <math.h>
#include <string>
#include <string.h>
#include <algorithm>
#include <stdio.h>
#include <deque>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef double db;
#define mp make_pair
#define rt return
#define rt0 return 0
#define rt1 return 1
const ll inf = 1000000007;
const int sz = 100100;

int n;
multiset<db> st[4];


int main() {
#ifdef _DEBUG
  freopen("in.txt", "r", stdin);   freopen("out.txt", "w", stdout);
#endif
  cin >> n;
  for(int i = 0; i < n; ++i){
    db x, y;
    cin >> x >> y;
    db d = sqrt(x * x + y * y);
    if(x > 0 && y > 0)
      st[0].insert(d);
    if(x < 0 && y > 0)
```

```
      st[1].insert(d);
    if(x < 0 && y < 0)
      st[2].insert(d);
    if(x > 0 && y < 0)
      st[3].insert(d);
  }
  int k = 0;
  int t = 0;
  while (!st[0].empty() || !st[1].empty() || !st[2].empty() || !st[3].empty()
){
    if(!st[k].empty()) {
      if((*st[k].begin()) - t < 0) {
        cout << "NO";
        rt0;
      }
      st[k].erase(st[k].begin());
    }



    ++k;
    k %= 4;
    ++t;
  }
  cout << "YES";
  return 0;
}
```

D. Amuz Deluxe

```
#include <iostream>
#include <stdlib.h>
#include <set>
#include <map>
#include <vector>
#include <cctype>
#include <queue>
#include <math.h>
#include <string>
#include <string.h>
#include <algorithm>
#include <stdio.h>
#include <deque>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef double db;
#define mp make_pair
#define rt return
#define rt0 return 0
#define rt1 return 1
const ll inf = 1000000007;
const int sz = 100100;

int n;
int k;
int ar[sz];
int main() {
#ifdef _DEBUG
  freopen("in.txt", "r", stdin);   freopen("out.txt", "w", stdout);
#endif
```

```
  cin >> n;

  for(int i = 0; i < n; ++i) {
    int t;
    cin >> t;
    if(k < 3) {
      ar[k++] = t;
      continue;
    }
    if(ar[k-1] == t) {
      ar[k++] = t;
      continue;
    }

    if(ar[k-1] == ar[k-2] && ar[k-1] == ar[k-3]) {
      int p = ar[k-1];
      while (k > 0 && ar[k - 1] == p) {
        --k;
      }
    }
    ar[k++] = t;

  }
  if(k >= 3 && ar[k-1] == ar[k-2] && ar[k-1] == ar[k-3]) {
      int p = ar[k-1];
      while (k > 0 && ar[k - 1] == p) {
        --k;
      }
    }
  cout << n - k;

  return 0;
}
```

E. Собеседование

```
#include <iostream>
#include <stdlib.h>
#include <set>
#include <map>
#include <vector>
#include <cctype>
#include <queue>
#include <math.h>
#include <string>
#include <string.h>
#include <algorithm>
#include <stdio.h>
#include <deque>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef double db;
#define mp make_pair
#define rt return
#define rt0 return 0
#define rt1 return 1
const ll inf = 1000000007;
const int sz = 10000;
bool prime[sz];
vector<int> p, cnt;

int n, m, k;
```

```
pair<int, vector<int>> fact(int arg){
	vector<int> ans;
	for(int i = 0; i < p.size() ; ++i){
		int t = 0;
		while (arg % p[i] == 0) {
			++t;
			arg /= p[i];
		}
		ans.push_back(t);
	}
	rt mp(arg, ans);
}



int main(){
#ifdef _DEBUG
	freopen("in.txt", "r", stdin);		freopen("out.txt", "w", stdout);
#endif
	for(int i = 2; i < sz; ++i) {
		if(prime[i])
			continue;
		p.push_back(i);
		for(int j = 2 * i; j < sz; j += i)
			prime[j] = 1;
	}
	cin >> n >> m >> k;
	for(int i = 0; i < p.size() ; ++i){
		int t = 0;
		ll del = p[i];
		while(del <= n) {
			t += n / del;
			del *= p[i];
		}


		cnt.push_back(t);
	}
	for(int i = 0; i < m ; ++i){
		bool b = 1;
		vector<int> ans;
		vector<int> ans2;
		map<int, int> ma;
		int t;
		cin >>t;
		pair<int, vector<int>> tmp = fact(t);
		if(tmp.first != 1 ) {
			ans2.push_back(tmp.first);
			if (ma.find(tmp.first) != ma.end())
				ma[tmp.first] = ma[tmp.first] + 1;
			else
				ma.insert(mp(tmp.first, 1));
		}
		ans = tmp.second;

		for(int j = 1; j < k && b; ++j) {
			cin >>t;
			tmp = fact(t);
			if(tmp.first != 1 ) {
				ans2.push_back(tmp.first);
				if (ma.find(tmp.first) != ma.end())
					ma[tmp.first] = ma[tmp.first] + 1;
				else
```

```
                                        ma.insert(mp(tmp.first, 1));
                        }
                        for(int it = 0; it < min(ans.size(),
tmp.second.size()); ++it)
                                ans[it] += tmp.second[it];
                }
                if(!b)
                        continue;
                int rs = inf;
                for(int j = 0; j < min(ans.size(), cnt.size()); ++j) {
                        if(ans[j] == 0)
                                continue;
                        rs = min(rs, cnt[j]/ans[j]);
                }
                for(int j = 0; j < ans2.size(); ++j) {
                        rs = min(rs, (n/ans2[j]) / ma[ans2[j]]);
                }
                cout << rs << endl;
        }


        return 0;
}
```

F. Шоколадка

```
#include <iostream>
#include <stdlib.h>
#include <set>
#include <map>
#include <vector>
#include <cctype>
#include <queue>
#include <math.h>
#include <string>
#include <string.h>
#include <algorithm>
#include <stdio.h>
#include <deque>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef double db;
#define mp make_pair
#define rt return
#define rt0 return 0
#define rt1 return 1
const ll inf = 1000000007;
const int sz = 100100;

int h, w;


int main() {
#ifdef _DEBUG
  freopen("in.txt", "r", stdin);   freopen("out.txt", "w", stdout);
#endif
  cin >> h >> w;
  if(h%2 == 0 || w % 2 == 0) {
    cout << 0;
    rt0;
  }
```

```
  int s = h * w;
  cout << min(h, w);
  return 0;
}
```

G. Супермаркет

```
#include <iostream>
#include <stdlib.h>
#include <set>
#include <map>
#include <vector>
#include <cctype>
#include <queue>
#include <math.h>
#include <string>
#include <string.h>
#include <algorithm>
#include <stdio.h>
#include <deque>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef double db;
#define mp make_pair
#define rt return
#define rt0 return 0
#define rt1 return 1
const ll inf = 1000000007;
const int sz = 100100;

int n, m, k;
ll ar[sz];
set<pair<ll, int>> st;

int main(){
#ifdef _DEBUG
  freopen("in.txt", "r", stdin);   freopen("out.txt", "w", stdout);
#endif
  cin >> n;
  for(int i = 0; i < n; ++i) {
    cin >> ar[i];
    st.insert(mp(ll(0), i));
  }
  cin >> m;
  for(int i = 0; i < m; ++i) {
    cin >> k;
    pair<ll, int> t = *st.begin();
    st.erase(st.begin());
    cout << t.second + 1 << " ";
    t.first += k * ar[t.second];
    st.insert(t);
  }

  return 0;
}
```

H. Разрядка и трансляции

```
#include <iostream>
#include <stdlib.h>
#include <set>
#include <map>
#include <vector>
#include <cctype>
#include <queue>
#include <math.h>
#include <string>
#include <string.h>
#include <algorithm>
#include <stdio.h>
#include <deque>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef double db;
#define mp make_pair
#define rt return
#define rt0 return 0
#define rt1 return 1
const ll inf = 1000000007;
const int sz = 100100;

int n, m, k;
ll ar[sz];
vector<pii> v, r;
multiset<int> st;
int ind = 0;
void add(int i, int t, int s) {
  while(!st.empty() && *st.begin() < s) {

    st.erase(st.begin());
  }

  while (ind < r.size() && r[ind].first <= t) {
    if(r[ind].second >= i) {
      st.insert(r[ind].first);
    }
    ++ind;
  }
}
int rs = 0;

int main(){
#ifdef _DEBUG
  freopen("in.txt", "r", stdin);   freopen("out.txt", "w", stdout);
#endif
  cin >> n;
  for(int i = 0; i < n; ++i) {
    int a, b;
    cin >> a >> b;
    v.push_back(mp(a, b));
  }
  cin >> m;
  int t;
  sort(v.begin(), v.end());
  for(int i = 0; i < n; ++i) {
    r.push_back(mp(v[i].second, i));
  }
```

```
  sort(r.begin(), r.end());
  for(int i = 0; i < v.size(); ++i){
    add(i, m + v[i].first, v[i].first);
    rs = max(rs, (int)st.size());
    if(!st.empty() && st.find(v[i].second) != st.end())
      st.erase(st.find(v[i].second));
  }
  cout << rs;
  return 0;
}
```

Решения задач участника Володин Вадим Евгеньевич (3 место)

A. Нормализация пути

```
#include <iostream>
#include <cstdio>
#include <vector>
#include <string>
#include <cstring>
#include <cmath>
#include <iomanip>
#include <algorithm>
#include <stack>
#include <queue>
#include <map>
#include <set>
#include <bitset>

using namespace std;

#define oo ((int)2e9)
#define eps 1e-6
#define forn(i, n) for (int i = 0; i < (n); i++)
#define rfor(i, n) for (int i = (n) - 1; i >= 0; i--)
#define it(v) v.begin(), v.end()
#define mp make_pair
#define pb push_back

typedef long long lint;
typedef long double ldouble;
typedef pair <int, int> pint;

int main(){
  ios_base::sync_with_stdio(false);
  cin.tie(NULL);
  string t; cin >> t;
  int now = 1;
  int tmp;
  vector <string> s;
  while ((tmp = t.find('/', now)) != string::npos){
    string strnow = t.substr(now, tmp - now);
    if (strnow == "."){}
    else if (strnow == ".."){ if (s.size() > 0) s.pop_back();}
    else
      s.push_back(strnow);
    now = tmp + 1;
  }
  cout << '/';
  for (int i = 0; i < s.size(); i++)
```

```
    cout << s[i] << '/';
}
```

C. Radar Defence

```
#include <iostream>
#include <cstdio>
#include <vector>
#include <string>
#include <cstring>
#include <cmath>
#include <iomanip>
#include <algorithm>
#include <stack>
#include <queue>
#include <map>
#include <set>
#include <bitset>

using namespace std;

#define oo ((int)2e9)
#define eps 1e-6
#define forn(i, n) for (int i = 0; i < (n); i++)
#define rfor(i, n) for (int i = (n) - 1; i >= 0; i--)
#define it(v) v.begin(), v.end()
#define mp make_pair
#define pb push_back

typedef long long lint;
typedef long double ldouble;
typedef pair <int, int> pint;

double g(int x, int y){
  double t = sqrt((double) x * x + y * y);
  return t;
}

int main(){
  ios_base::sync_with_stdio(false);
  cin.tie(NULL);
  int n; cin >> n;
  vector <double> f[4];
  for (int i = 0; i < n; i++){
    int x, y;
    cin >> x >> y;
    if (x >= 0 && y >= 0)
      f[0].push_back(g(x, y));
    else if (y >= 0)
      f[1].push_back(g(x, y));
    else if (x >= 0)
      f[3].push_back(g(x, y));
    else
      f[2].push_back(g(x, y));
  }
  forn (i, 4)
    sort(it(f[i]));
  int now = 0;
  int ended = 0;
  vector <int> ind(4);
  while (true){
    int p = now % 4;
    if (ind[p] == f[p].size()){
      ended++;
```

```
      ind[p] = -1;
    }
    if (ended == 4)
      break;
    if (ind[p] == -1){
      now++;
      continue;
    }
    if (now > f[p][ind[p]]){
      cout << "NO";
      return 0;
    }
    ind[p]++;
    now++;
  }
  cout << "YES";
}
```

D. Amuz Deluxe

```
#include <iostream>
#include <cstdio>
#include <vector>
#include <string>
#include <cstring>
#include <cmath>
#include <iomanip>
#include <algorithm>
#include <stack>
#include <queue>
#include <map>
#include <set>
#include <bitset>

using namespace std;

#define oo ((int)2e9)
#define eps 1e-6
#define forn(i, n) for (int i = 0; i < (n); i++)
#define rfor(i, n) for (int i = (n) - 1; i >= 0; i--)
#define it(v) v.begin(), v.end()
#define mp make_pair
#define pb push_back

typedef long long lint;
typedef long double ldouble;
typedef pair <int, int> pint;

int main(){
  ios_base::sync_with_stdio(false);
  cin.tie(NULL);
  int n; cin >> n;
  vector <int> a(n);
  forn (i, n)
    cin >> a[i];
  int left = -1, right;
  for (int i = 0; i + 2 < n; i++)
    if (a[i] == a[i + 1] && a[i] == a[i + 2])
      left = i;
  if (left == -1){
    cout << 0;
    return 0;
  }
  int res = 3;
```

```
  right = left + 2;
  left--; right++;
  while (left >= 0 && right < n){
    if (a[left] == a[right]){
      int ths = a[left];
      int now = 0;
      while (left >= 0 && a[left] == ths){
        left--;
        now++;
      }
      while (right < n && a[right] == ths){
        right++;
        now++;
      }
      if (now >= 3)
        res += now;
      else
        break;
    }
    else
      break;
  }
  cout << res;
}
```

E. Собеседование

```
#include <iostream>
#include <cstdio>
#include <vector>
#include <string>
#include <cstring>
#include <cmath>
#include <iomanip>
#include <algorithm>
#include <stack>
#include <queue>
#include <map>
#include <set>
#include <bitset>

using namespace std;

#define oo ((int)2e9)
#define eps 1e-6
#define forn(i, n) for (int i = 0; i < (n); i++)
#define rfor(i, n) for (int i = (n) - 1; i >= 0; i--)
#define it(v) v.begin(), v.end()
#define mp make_pair
#define pb push_back

typedef long long lint;
typedef long double ldouble;
typedef pair <int, int> pint;

int main(){
  ios_base::sync_with_stdio(false);
  cin.tie(NULL);

  lint n, m, k; cin >> n >> m >> k;

  map <lint, int> s;

  for (lint i = 2; i * i <= n; i++){
```

```
    lint t = i;
    while (t <= n){
      s[i] += n / t;
      t *= i;
    }
  }

  forn(i, m){
    map <lint, int> now;
    forn(j, k){
      lint a; cin >> a;
      lint b = a;
      for (int i = 2; i * i <= a; i++)
        while (b % i == 0){
          now[i]++;
          b /= i;
        }
      if (b != 1)
        now[b]++;
    }
    int res = -1;
    for (map<lint, int>::iterator t = now.begin(); t != now.end(); ++t){
      lint ths = t->first;
      int cnt;
      if (ths * ths <= n)
        cnt = s[ths] / now[ths];
      else
        cnt = (n / ths) / now[ths];
      if (res == -1 || cnt < res)
        res = cnt;
    }
    cout << res << '\n';
  }
  int b;
}
```

F. Шоколадка

```
#include <iostream>
#include <cstdio>
#include <vector>
#include <string>
#include <cstring>
#include <cmath>
#include <iomanip>
#include <algorithm>
#include <stack>
#include <queue>
#include <map>
#include <set>
#include <bitset>

using namespace std;

#define oo ((int)2e9)
#define eps 1e-6
#define forn(i, n) for (int i = 0; i < (n); i++)
#define it(v) v.begin(), v.end()
#define mp make_pair
#define pb push_back

typedef long long lint;
typedef long double ldouble;
typedef pair <int, int> pint;
```

```
int main(){
  ios_base::sync_with_stdio(false);
  cin.tie(NULL);
  int w, h; cin >> w >> h;
  if (w % 2 == 0 || h % 2 == 0)
    cout << 0;
  else
    cout << min(w, h);
}
```

G. Супермаркет

```
#include <iostream>
#include <cstdio>
#include <vector>
#include <string>
#include <cstring>
#include <cmath>
#include <iomanip>
#include <algorithm>
#include <stack>
#include <queue>
#include <map>
#include <set>
#include <bitset>

using namespace std;

#define oo ((int)2e9)
#define eps 1e-6
#define forn(i, n) for (int i = 0; i < (n); i++)
#define it(v) v.begin(), v.end()
#define mp make_pair
#define pb push_back

typedef long long lint;
typedef long double ldouble;
typedef pair <int, int> pint;

struct seller{
  lint will, index, t;
};
bool operator <(const seller& a, const seller& b){
  if (a.will != b.will)
    return a.will < b.will;
  return a.index < b.index;
}
int main(){
  ios_base::sync_with_stdio(false);
  cin.tie(NULL);
  set <seller> sellers;
  int n; cin >> n;
  forn(i, n){
    int t; cin >> t;
    seller s;
    s.will = 0;
    s.index = i;
    s.t = t;
    sellers.insert(s);
  }
  int m; cin >> m;
  forn(i, m){
    lint a; cin >> a;
```

```
    seller t = *sellers.begin();
    sellers.erase(sellers.begin());
    t.will += t.t * a;
    sellers.insert(t);
    cout << t.index + 1 << ' ';
  }
}
```

H. Разрядка и трансляции

```
#include <iostream>
#include <cstdio>
#include <vector>
#include <string>
#include <cstring>
#include <cmath>
#include <iomanip>
#include <algorithm>
#include <stack>
#include <queue>
#include <map>
#include <set>
#include <bitset>

using namespace std;

#define oo ((int)2e9)
#define eps 1e-6
#define forn(i, n) for (int i = 0; i < (n); i++)
#define rfor(i, n) for (int i = (n) - 1; i >= 0; i--)
#define it(v) v.begin(), v.end()
#define mp make_pair
#define pb push_back

typedef long long lint;
typedef long double ldouble;
typedef pair <int, int> pint;

int main(){
  ios_base::sync_with_stdio(false);
  cin.tie(NULL);
  int n; cin >> n;

  vector <pint> rl(n);
  forn(i, n)
    cin >> rl[i].second >> rl[i].first;
  sort(it(rl));
  int m; cin >> m;
  int mx = 0;
  multiset<int> now;
  forn (i, n){
     now.insert(rl[i].second);
     while (now.size() && *now.begin() + m < rl[i].first)
       now.erase(now.begin());
     mx = max(mx, (int)now.size());
  }
  cout << mx;
}

/*
3
2 4
3 6
```

```
4 7

3
1 4
3 4
2 4
3
*/
```