

Задача А Две стены

Имя входного файла: *input.txt*
Имя выходного файла: *output.txt*
Максимальное время работы на одном тесте: *10 секунд*

У Пети есть набор из N кирпичиков. Каждый кирпичик полностью окрашен в один из K цветов, i -й кирпичик имеет размер $1 \times 1 \times L_i$.

Петя знает, что он может построить из кирпичиков прямоугольную стену толщиной 1 и высотой K , причем первый горизонтальный слой кирпичиков в стене будет первого цвета, второй – второго и т. д. Теперь Петя хочет узнать, может ли он из своего набора построить две прямоугольные стены, обладающие тем же свойством. Помогите ему выяснить это.

Формат входных данных

На первой строке входного файла находятся числа N и K ($1 \leq N \leq 5000$, $1 \leq K \leq 100$). Следующие N строк содержат описание Петиних кирпичиков: сначала длина L_i , затем номер цвета C_i ($1 \leq L_i \leq 100$, $1 \leq C_i \leq K$). Известно, что у Пети не более 50 кирпичиков каждого цвета.

Формат выходных данных

Выведите на первой строке выходного файла YES, если Петя сможет построить из своих кирпичиков две прямоугольные стены высоты K , j -й слой кирпичиков в каждой из которых будет j -го цвета, и NO в противном случае. В случае положительного ответа, выведите на второй строке в произвольном порядке номера кирпичиков, из которых следует построить первую стену (кирпичики нумеруются в том порядке, в котором они заданы во входном файле, начиная с 1). Если решений несколько, можно выдать любое из них.

Примеры

input.txt	output.txt
5 2 2 1 2 1 2 1 3 2 3 2	NO
2 1 1 1 3 1	YES 1
9 3 7 3 1 1 3 2 2 1 3 2 3 1 4 2 4 1 3 3	YES 2 3 4 9

Задача В Блокада

Имя входного файла: *input.txt*
Имя выходного файла: *output.txt*
Максимальное время работы на одном тесте: *2 секунды*

Государство Флатландия представляет собой прямоугольник размером $M \times N$, состоящий из единичных квадратов. Флатландия разделена на K провинций ($2 \leq K \leq 100$). Каждая провинция представляет собой связное множество квадратов, т.е. из каждой точки провинции можно дойти до любой другой ее точки, при этом разрешается переходить с квадрата на квадрат, только если они имеют общую сторону (общей вершины недостаточно). Во Флатландии нет точки, которая граничила бы более чем с тремя провинциями (т.е. четыре квадрата, имеющие общую вершину, не могут принадлежать четырем разным провинциям).

Каждая провинция имеет свой символ. Столица Флатландии находится в провинции, которой принадлежит символ A (заглавная латинская буква A). Провинция называется пограничной, если она содержит граничные квадраты. Провинция, в которой находится столица Флатландии, не является пограничной.

Король соседнего с Флатландией королевства Ректиландия решил завоевать Флатландию. Для этого он хочет захватить столицу Флатландии. Однако он знает, что сил его армии недостаточно, чтобы сделать это сразу. Поэтому сначала он хочет окружить столичную провинцию, чтобы ослабить силы противника долгой блокадой, а потом захватить столицу.

Чтобы окружить провинцию, требуется захватить все провинции, с которыми она граничит. Две провинции граничат, если существует два квадрата, имеющие общую сторону, один из которых принадлежит первой из них, а другой – второй. Чтобы захватить провинцию, надо чтобы выполнялось одно из двух условий: либо она пограничная, либо граничит с какой-либо уже захваченной провинцией.

Чтобы сберечь силы своей армии, король Ректиландии хочет установить блокаду столичной провинции, захватив как можно меньше провинций. Помогите ему выяснить, сколько провинций требуется захватить. Захватывать столичную провинцию нельзя, поскольку для этого сил армии Ректиландии пока недостаточно.

Формат входных данных

Первая строка содержит M и N ($3 \leq M, N \leq 200$). Следующие M строк содержат N символов каждая и задают карту Флатландии. Символ, находящийся в $i+1$ -й строке входного файла на j -м месте, представляет собой символ провинции, которой принадлежит квадратик (i, j) . Все символы имеют ASCII-код больше 32 (пробела).

Формат выходных данных

Выведите в выходной файл единственное число – количество провинций, которые требуется захватить. Если установить блокаду невозможно, выведите “–1”.

Примеры

<i>input.txt</i>	<i>output.txt</i>
5 6 BBBBBZ BBBBBZ BCAbbZ BDDDbZ 33333Z	4

Задача С Электронные часы

Имена входных файлов:

digits.txt, input.txt

Имя выходного файла:

output.txt

Максимальное время работы на одном тесте:

2 секунды

Циферблат новых электронных часов, установленных на главном здании офиса фирмы *Macrohard*, состоит из 4 прямоугольных панелей, каждая из которых состоит из 6 рядов по 5 лампочек в каждом. Первые две панели отображают цифры, из которых складываются часы, а следующие две – минуты. (Если сейчас меньше 10 часов, первая панель отображает 0).

К сожалению, лампочки, установленные на панелях, были произведены компанией *Sveta.Net*, которая известна своим принципом "*раньше перегорит – больше спрос*", вследствие чего на следующий день люди, проходя мимо офиса компании, видели лишь некоторое подобие цифр, поскольку некоторые лампочки больше не горели.

Петя живет в доме, стоящем прямо напротив офиса компании *Macrohard*. В первый день после установки часов он зарисовал у себя в блокноте, как выглядят все цифры на панелях (панели однотипные, поэтому одна и та же цифра на различных панелях выглядит одинаково). Теперь Петя хочет узнать, можно ли по текущему изображению на часах однозначно определить, сколько сейчас времени. Помогите ему!

Формат входных данных

При тестировании этой задачи в каталоге, который будет текущим, когда будет запущена Ваша программа, будет находиться два файла. Файл *digits.txt* содержит 6 строк по 50 символов в каждой. Он будет одинаковым для всех тестов и будет совпадать с приведенным в примере. Вы также можете найти этот файл в каталоге `o:\common`. Содержимое файла *digits.txt* задает правильное написание цифр на панелях (первый прямоугольник 6×5 символов задает число 0, следующий – 1, и т. д. до 9). Не горящая лампочка обозначается символом "." (точка), а горящая – "#" (диз).

Входной файл *input.txt* содержит 6 строк по 20 символов в каждой – текущее изображение на часах. Первый прямоугольник 6×5 задает первую панель, следующий – вторую, следующий – третью и последний – четвертую.

Формат выходных данных

Если можно точно определить время, которое сейчас отображается на часах, выведите это время в формате `hh:mm`. Если время нельзя определить однозначно, выведите `AMBIGUITY`. Если же в часах точно сломалось еще что-то, например центральный процессор, который управляет лампочками, выведите `ERROR`.

Примеры

digits.txt	
<pre>..##.....#..##.####.#...#.....####.####.####.####. .#...#...#.#...#...#...#...#...#...#...#...#...#...#...# .#...#...#.#...#...#.#...#...#...#...#...#...#...#...#...# .#...#...#.#...#...#.#...#...#...#...#...#...#...#...#...# .#...#...#.#...#...#.#...#...#...#...#...#...#...#...#...# ..##.....#..##.####.#...#.....####.####.####.####.</pre>	
input.txt	output.txt
<pre>..##.####.#...#...#...#...#...#...#...#...#...#...#...# ...#...#...#...#...#...#...#...#...#...#...#...#...#...# ...#...#...#...#...#...#...#...#...#...#...#...#...#...# ...#...#...#...#...#...#...#...#...#...#...#...#...#...# ...#...#...#...#...#...#...#...#...#...#...#...#...#...# ####.#...#...#...#...#...#...#...#...#...#...#...#...#...</pre>	23:45
<pre>#...#...#...#...#...#...#...#...#...#...#...#...#...# .#...#...#...#...#...#...#...#...#...#...#...#...#...# .#...#...#...#...#...#...#...#...#...#...#...#...#...# .#...#...#...#...#...#...#...#...#...#...#...#...#...# .#...#...#...#...#...#...#...#...#...#...#...#...#...# ...#...#...#...#...#...#...#...#...#...#...#...#...#...</pre>	ERROR
<pre>..##.####.#...#...#...#...#...#...#...#...#...#...#...# .#...#...#...#...#...#...#...#...#...#...#...#...#...# .#...#...#...#...#...#...#...#...#...#...#...#...#...# .#...#...#...#...#...#...#...#...#...#...#...#...#...# .#...#...#...#...#...#...#...#...#...#...#...#...#...# ...#...#...#...#...#...#...#...#...#...#...#...#...#...</pre>	AMBIGUITY

Задача D Сравнение URL

Имя входного файла: *input.txt*
Имя выходного файла: *output.txt*
Максимальное время работы на одном тесте: *2 секунды*

Для идентификации ресурсов в сети Internet используются URL (Uniform Resource Locator). URL состоит из нескольких элементов: *протокол*, *хост*, *порт*, *путь*, *файл* и *секция*. Некоторые элементы URL могут быть опущены. Рассмотрим упрощенный формат URL:

[*протокол* : //] *хост* [: *порт*] [*путь* / [*файл* [*#секция*]]]

Заключенные в квадратные скобки элементы могут быть опущены, т.е. например, можно не указать протокол или секцию. Но, например, если указан файл, то обязательно должен быть указан путь. Регистр букв в элементах URL не важен.

Рассмотрим кратко все элементы URL:

Протокол – это способ доступа к файлу, URL с разными протоколами и одинаковыми остальными элементами могут указывать на различные ресурсы.

Хост и *порт* – это имя некоторого сервера в сети и способ доступа к нему (порт – натуральное число, не превосходящее 65535).

Путь представляет собой путь к файлу, содержащему запрашиваемый ресурс, от некоторого каталога на сервере, который называется *корневым*. При этом для разделения имен каталогов используется символ "/". Путь, если он не пуст, всегда начинается с символа "/". Специальное обозначение "." соответствует самому каталогу, ".." – родительскому каталогу.

Файл – это файл, содержащий запрашиваемый ресурс.

Наконец, файл может быть разбит на *секции* каким-либо способом и можно указать, к какой именно секции вы хотите обратиться.

Различные символы в URL могут быть заменены своими шестнадцатеричными ASCII-кодами с помощью символа %, например *a* = %41, *Z* = %5A. В коде всегда используется ровно две шестнадцатеричные цифры.

Некоторые символы могут встречаться в элементах URL только как шестнадцатеричные коды – все символы, кроме букв латинского алфавита, цифр и символов "." и "-", а некоторые не могут встречаться вообще: "\", "#", "*", "@", "%", "?", ":", ";", а также символы с ASCII-кодом меньше %20. Символ "/" может встречаться в элементах URL только в пути для разделения входящих в него каталогов. Имя файла не может состоять только из точек.

Рассмотрим примеры URL:

```
http://neerc.ifmo.ru/school
ftp://somewhere.net:1234/pub/files/coolgame.zip
nobody.nowhere.net/some%20dir/some%20file#some%20info
```

Ваша цель в этой задаче – помочь разработчикам web-сервера. Для web-сервера отсутствующие части URL имеют следующие значения по умолчанию:

Протокол	http
Порт	80
Путь	<i>пустая строка</i>
Файл	index.html
Секция	<i>пустая строка</i>

Различные как строки URL могут указывать на один и тот же ресурс, например следующие три URL:

```
neerc.ifmo.ru
http://neerc.ifmo.ru:80/index.html#
Http://NEERC.IFMO.Ru/Dir/../../../../
```

Для разграничения доступа к ресурсам необходимо уметь определять, указывают ли два различных URL на один и тот же ресурс. Помогите разработчикам написать соответствующую проверку.

Формат входных данных

Входной файл состоит из двух строк, каждая из них содержит URL. Оба URL удовлетворяют формату, приведенному в условии этой задачи. Длина каждого URL не превосходит 200 символов. Гарантируется, что ни один из промежуточных каталогов на пути к ресурсу не лежит выше корневого каталога (т.е. не может встретиться, например, URL `http://somewhere.com/../../dir/index.html`) а также, что имена всех каталогов состоят по крайней мере из одного символа (два символа "/" не могут идти подряд в любом месте, кроме как непосредственно после двоеточия после имени протокола).

Формат выходных данных

Выведите YES в выходной файл, если оба URL, приведенные во входном файле, указывают на один и тот же ресурс и NO в противном случае.

Примеры

input.txt	output.txt
<code>http://neerc.ifmo.ru:80/index.html# neerc.ifmo.ru</code>	YES
<code>neerc.ifmo.ru/dir/../../school NEERC.IFMO.RU/./SCHOOL</code>	YES
<code>neerc.ifmo.ru %6E%65%65%72%63%2E%69%66%6D%6F%2E%72%75</code>	YES
<code>nowhere.com somewhere.com</code>	NO
<code>http://neerc.ifmo.ru ftp://neerc.ifmo.ru</code>	NO
<code>neerc.ifmo.ru/index.htm neerc.ifmo.ru/index.html</code>	NO
<code>neerc.ifmo.ru:80 neerc.ifmo.ru:8080/index.html</code>	NO
<code>http://somewhere.com somewhere.com/index.html#a</code>	NO

Задача Е Триангуляция Делоне

Имя входного файла:

input.txt

Имя выходного файла:

output.txt

Максимальное время работы на одном тесте:

10 секунд

Триангуляцией некоторого набора точек на плоскости называется набор невырожденных треугольников, удовлетворяющий следующим свойствам:

1. Вершинами треугольников являются только точки исходного набора. Каждая точка исходного набора является вершиной хотя бы одного треугольника.
2. Два различных треугольника либо не имеют общих точек, либо имеют общую вершину, либо имеют общую сторону (но площадь их пересечения всегда равна 0).
3. Любая точка, лежащая внутри выпуклой оболочки исходного набора точек, принадлежит хотя бы одному треугольнику (она может принадлежать нескольким треугольникам, если является их общей вершиной или принадлежит их общей стороне). (Выпуклой оболочкой некоторого набора точек называется наименьший выпуклый многоугольник, содержащий все эти точки).

Триангуляция называется триангуляцией Делоне, если кроме того для нее выполняется следующее условие:

4. Внутри окружности, описанной около любого треугольника из триангуляции, не лежит ни одна из исходных точек (точки могут лежать на окружности, в частности на ней, очевидно, лежат вершины рассматриваемого треугольника)

Для заданного набора точек найдите количество его триангуляций Делоне (две триангуляции считаются различными, если они отличаются хотя бы одним треугольником)

Формат входных данных

На первой строке входного файла находится число N – количество точек ($3 \leq N \leq 30$) исходного набора. Следующие N строк содержат по одной паре вещественных чисел – координаты соответствующей точки. Никакие три точки не лежат на одной прямой.

Формат выходных данных

Выведите в выходной файл количество различных триангуляций Делоне указанного набора точек.

Пример

<i>input.txt</i>	<i>output.txt</i>
4 0.0 0.0 1.0 0.0 0.0 1.0 1.0 1.0	2

Задача F Яблоко от яблони...

Имя входного файла:

input.txt

Имя выходного файла:

output.txt

Максимальное время работы на одном тесте:

2 секунды

У Пети в саду растет яблоня. Воодушевленный историей об Исааке Ньютоне, который, как известно, открыл закон всемирного тяготения после того, как ему на голову упало яблоко, Петя с целью повысить свою успеваемость по физике часто сидит под яблоней.

Однако, поскольку по физике у Пети твердая тройка, яблоки с его яблони падают следующим образом. В какой-то момент одно из яблок отрывается от ветки, на которой оно висит, и начинает падать строго вниз. Если в некоторый момент оно задевает другое яблоко, то то тоже отрывается от своей ветки и начинает падать вниз, при этом первое яблоко не меняет направление своего падения. Вообще, если любое падающее яблоко заденет другое яблоко на своем пути, то оно также начнет падать.

Таким образом, в любой момент каждое яблоко либо висит на ветке, либо падает строго вниз, причем все яблоки кроме первого, чтобы начать падать, должны сначала соприкоснуться с каким-либо другим падающим яблоком.

Выясните, какие яблоки упадут с Петиной яблони.

Формат входных данных

Первая строка входного файла содержит N - количество яблок на петиной яблоне ($1 \leq N \leq 200$). Следующие N строк содержат описания яблок. Будем считать все яблоки шарами. Каждое яблоко задается координатами своей самой верхней точки (той, где оно исходно прикреплено к дереву, длиной черенка пренебрежем) x_i , y_i и z_i и радиусом r_i ($-10000 \leq x_i, y_i, z_i \leq 10000$, $1 \leq r_i \leq 10000$, все числа целые). Гарантируется, что изначально никакие яблоки не пересекаются (даже не соприкасаются). Ось OZ направлена вверх.

Формат выходных данных

Выведите на первой строке выходного файла количество яблок, которые упадут с яблони, если начнет падать первое яблоко. На следующей строке выведите номера упавших яблок. Яблоки нумеруются, начиная с 1, в том порядке, в котором они заданы во входном файле.

Примеры

<i>input.txt</i>	<i>output.txt</i>
4 0 0 10 4 5 0 3 1 -7 4 7 1 0 1 2 6	3 1 2 4

Задача G Контрольный блок

Имя входного файла:

input.txt

Имя выходного файла:

output.txt

Максимальное время работы на одном тесте:

2 секунды

Фирма *Macrohard* разработала новый протокол обмена данными по сети. Каждый блок данных при этом обмене состоит из N чисел в диапазоне от 0 до $M-1$ включительно. Чтобы повысить надежность передачи, вместе с блоком данных пересылается контрольный блок такой же длины.

Предположим, что исходный блок состоит из чисел $a_1, a_2, \dots, a_N$. Тогда, контрольный блок состоит из чисел $b_1, b_2, \dots, b_N$, из диапазона от 0 до $M-1$ включительно, таких что выполняются следующие равенства: $b_1 = (a_N + b_N) \bmod M$, $b_2 = (a_1 + b_1) \bmod M$, $b_3 = (a_2 + b_2) \bmod M$, ..., $b_N = (a_{N-1} + b_{N-1}) \bmod M$ (обозначение $X \bmod M$ обозначает остаток от деления X на M , например, $7 \bmod 4 = 3$, $6 \bmod 2 = 0$).

Блоки данных, для которых нельзя построить контрольный блок, удовлетворяющий указанному свойству, считаются подозрительными и их передача по сети не разрешается.

Ваня хочет поступить на работу программистом в фирму *Macrohard*, и в качестве вступительного задания ему поручили написать процедуру построения контрольного блока для заданного блока данных. Помогите ему!

Формат входных данных

Первая строка входного файла содержит числа N и M ($1 \leq N \leq 1000$, $2 \leq M \leq 10^9$). Следующая строка содержит блок данных, для которого следует построить контрольный блок, числа разделены пробелами.

Формат выходных данных

На первой строке выходного файла выведите YES, если для данного блока данных можно построить контрольный блок и NO, если нельзя. В случае, если контрольный блок построить можно, на второй строке выведите контрольный блок. Числа разделяйте пробелами. Если решений несколько, можно выдать любое из них.

Примеры

<i>input.txt</i>	<i>output.txt</i>
4 3 1 0 0 2	YES 1 2 2 2
4 3 1 1 1 1	NO

Задача Н Куб

Имя входного файла:

input.txt

Имя выходного файла:

output.txt

Максимальное время работы на одном тесте:

10 секунд

Петя склеил из N^3 единичных кубиков большой куб размером $N \times N \times N$. Устав от этой сложной работы, он отправился спать, а утром, проснувшись, с ужасом обнаружил, что его младший брат Ваня K раз проткнул куб спицей.

При этом Ваня действовал очень аккуратно, каждый раз установив конец спицы точно в центр грани какого-нибудь граничного единичного кубика, он протыкал куб параллельно соответствующей оси координат, при этом целый ряд из N кубиков оказывался испорчен.

Немного успокоившись после этого тяжелого потрясения, Петя заинтересовался, сколько кубиков в его творении осталось неповрежденными. Помогите ему ответить на этот сложный вопрос.

Формат входных данных

На первой строке входного файла находятся числа N и K ($1 \leq N \leq 1000$, $0 \leq K \leq 150$). Следующие K строк описывают Ванины преступные действия. Каждая строка содержит три числа — два из них представляют собой соответствующие координаты всех кубиков, проткнутых спицей, а третье, соответствующее координате, в направлении которой был проткнут куб, равно 0. Например, если $N = 3$, тройка $(1, 0, 3)$ означает, что спицей были проткнуты кубики $(1, 1, 3)$, $(1, 2, 3)$ и $(1, 3, 3)$. Все координаты лежат в пределах от 1 до N . Известно, что Ваня никакое действие не выполнял два раза (т.е. никакая тройка не встретится во входном файле дважды).

Формат выходных данных

Выведите в выходной файл единственное число — количество неповрежденных кубиков.

Пример

<i>input.txt</i>	<i>output.txt</i>
3 2 2 2 0 2 0 1	22