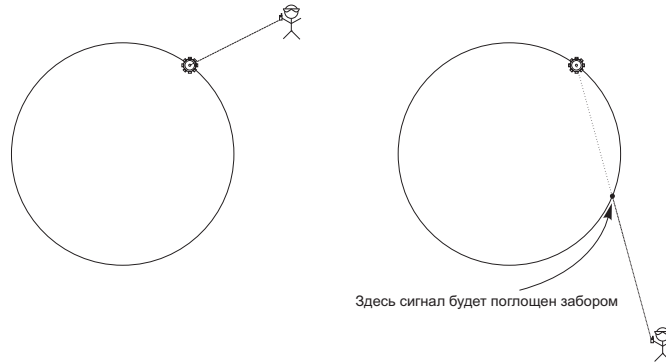


Задача А. Шпион

Ограничение по времени: 2 секунды
 Ограничение по памяти: 64 Мб

Секретный объект обнесен забором, выполненным в форме окружности радиуса r с центром в точке $(0, 0)$. Шпион, пытающийся разведать точное местоположение секретного объекта, для преодоления забора установил бомбу точно на заборе в точке (a_x, a_y) . Бомба активизируется дистанционным взрывателем, сигнал от которого распространяется по прямой, причём качество его приёма не зависит от пройденного расстояния. Из секретных источников известно, что забор полностью поглощает проходящий сквозь него сигнал (на сигнал, проходящий по касательной, забор не влияет). Шпион, находясь вне области, окруженной забором, в точке (b_x, b_y) , решил взорвать бомбу. Помогите ему узнать, дойдет ли сигнал до бомбы или будет поглощен.



Формат входных данных

В первой строке входного файла записано число r ($r > 0$). Во второй строке входного файла содержатся два числа (a_x, a_y) . В третьей строке входного файла содержатся два числа (b_x, b_y) . Все числа во входном файле целые и не превышают по модулю 10^3 .

Формат выходных данных

В единственной строке выходного файла выведите слово YES, если сигнал дойдет до бомбы, и NO в противном случае.

Примеры

ВВОД	ВЫВОД
1 1 0 2 0	YES
5 3 4 5 -1	NO
5 -3 -4 0 -10	YES

Задача В. Наилучшая расческа

Ограничение по времени: 5 секунд

Ограничение по памяти: 64 Мб

После того, как задачи нахождения в числовой матрице наибольших квадратов и прямоугольников из нулей стали стандартными, появилась новая задача, обобщающая все предыдущие. Дана матрица из нулей и единиц. Необходимо найти наилучшую “расческу”, которую можно построить на нулевых ячейках этой матрицы (наилучшей называется расческа, которая занимает наибольшее количество ячеек данной матрицы). Звеном расчески называется непустая прямоугольная область из нулей в матрице, то есть некоторая подматрица ненулевой площади, состоящая только из нулей. k -расческой называется фигура, получаемая соединением по вертикали k ($0 \leq k \leq m$) звеньев. У всех звеньев должна совпадать x -координата левой стороны, соседние звенья должны иметь разную ширину, а все звенья вместе должны образовывать цельную фигуру, то есть должны быть связаны. При этом 0-расческа существует всегда и имеет размер 0.

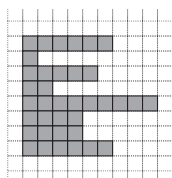


рис. 1

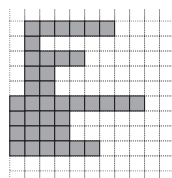


рис. 2

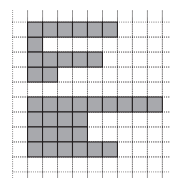


рис. 3

Фигура на рисунке 1 является правильной 7-расческой. Хотя фигура занимает 8 строк, она является 7-расческой, так как на второй и третьей снизу строках находится одно и то же звено (соседние звенья обязательно должны быть разной ширины, но звено может быть высотой более 1 строки). Фигура на рисунке 2 не является правильной расческой, так как левая x -координата звеньев не совпадает. Фигура на рисунке 3 также не является правильной расческой, так как не является связной фигурой. Можно заметить, что наилучший прямоугольник это то же самое что наилучшая 1-расческа (расческа высоты 1 звено). Очевидно, что для каждой длины k в матрице можно либо найти наилучшую расческу, либо заявить, что ее нет (то есть наилучшая расческа имеет размер 0). Например, на рисунке 4 обозначена лучшая 3-расческа (знаком X обозначены ненулевые элементы в матрице).

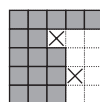


рис. 4

Ваша задача — найти в данной матрице наилучшую расческу среди всех k -расчесок матрицы (для всех k от 0 до m).

Формат входных данных

В первой строке входного файла даны два целых числа — высота ($1 \leq m \leq 2000$) и ширина ($1 \leq n \leq 2000$) матрицы соответственно. Далее, в m строках дано по n чисел через пробел — исходная матрица.

Формат выходных данных

В выходной файл необходимо вывести размер наибольшей расчески среди всех k -расчесок, которую можно построить на нулевых ячейках входной матрицы.

Примеры

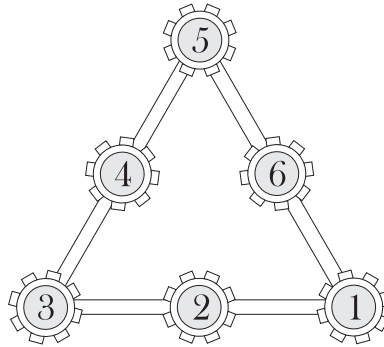
ВВОД	ВЫВОД
3 4 0 0 1 0 0 1 0 0 0 0 0 0	7
5 5 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0	20

Задача С. Крепость

Ограничение по времени: 2 секунды
 Ограничение по памяти: 64 Мб

В 2123 году на территории Ленинградской Области археологи обнаружили останки старинной крепости. К сожалению, некоторые фрагменты крепости не сохранились. Археологи точно знают, что крепость содержала шесть башен, три из которых являются вершинами треугольника, а остальные три — серединами сторон этого треугольника. Известно точное расположение только некоторых башен. Ваша задача определить расположение всех башен.

На рисунке приведен возможный вид крепости сверху:



Формат входных данных

Входной файл содержит шесть строк, каждая строка представляет собой описание башни. Если расположение башни известно, то строка содержит два целых числа, разделенных одним пробелом, в противном случае два знака вопроса ('?'), разделенных одним пробелом. Башни даны в порядке обхода, начиная с любой угловой башни.

Формат выходных данных

В выходной файл необходимо вывести **IMPOSSIBLE**, если однозначно восстановить расположение всех башен невозможно, в противном случае вывести в первой строке **POSSIBLE**, а в следующих шести строках расположение башен, в таком же порядке как во входном файле. Числа необходимо вывести по крайней мере с двумя знаками после точки.

Примеры

ВВОД	ВЫВОД
0 0	POSSIBLE
0 1	0.00 0.00
? ?	0.00 1.00
? ?	0.00 2.00
2 2	1.00 2.00
? ?	2.00 2.00
	1.00 1.00
0 1	IMPOSSIBLE
? ?	
? ?	
? ?	
-2 -1	
-1 0	

Задача D. Маленький шахматный Ним

Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

В шахматной стране в последнее время стала очень популярна игра Ним. Правила игры просты. Перед началом игры на стол выкладываются несколько кучек камней. Два игрока ходят по очереди и за каждый ход берут из одной любой кучки произвольное число камней. Игрок, который берет последний камень из последней оставшейся кучки — проигрывает.

Черный и белый короли тоже решили сыграть в Ним, но игра оказалась слишком сложной, поэтому они решили немного изменить правила: камни теперь можно брать не из любой кучки, а только из такой, в которой содержится минимальное число камней.

После нескольких партий обнаружилось, что черный король очень хорошо освоил эту игру и каждый раз ходит наилучшим образом, то есть если у черного короля есть хотя бы один ход, ведущий к победе, то он его и делает. Таким образом, белый король стал подозревать, что исход каждой партии можно определить по начальной позиции в игре.

Теперь он хочет, чтобы вы, как главный мудрец шахматной страны, помогли ему определить по количеству камней в каждой кучке, может ли он выиграть, и если может, то сколько камней ему нужно взять из минимальной кучки для того, чтобы сохранить возможность победы.

Формат входных данных

В первой строке входного файла записано целое число n — количество кучек ($1 \leq n \leq 100$). Во второй строке входного файла записано n целых чисел b_i ($1 \leq b_i \leq 1000$) — количество камней в i -ой кучке.

Формат выходных данных

Если белый король может выиграть при наилучшей игре черного короля, то в первую строку выходного файла выведите слово YES, а во вторую строку — целое число s , которые определяет, сколько камней необходимо взять белому королю из минимальной кучки на первом ходе.

Если же белый король не может выиграть, то в первую строку файла выведите NO.

Примеры

ввод	вывод
2	YES
2 3	1
1	NO
1	

Задача Е. Сколько же переменных?

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 Мб

Маленький мальчик Петя очень любит программировать на языке Паскаль. К сожалению, в последнее время он стал замечать, что использует слишком много переменных. Так много, что сам не может толком сосчитать, сколько же в каждой программе их используется.

Поэтому он просит вас написать программу, которая поможет ему в этом нелегком деле.

Так как Петя еще не очень хорошо знает Паскаль, то использует он только четыре стандартных типа: `boolean`, `char`, `double`, `integer`.

Формат входных данных

Во входном файле содержится несколько строк с объявлениями переменных.

Каждая строка входного файла является объявлением переменных одного типа.

Все объявления являются корректными с точки зрения синтаксиса языка Паскаль, названия переменных не повторяются, при объявлении используются только стандартные типы: `boolean`, `char`, `double`, `integer`.

Формально каждое объявление имеет следующий вид:

```
<идентификатор> {',' <идентификатор>} ':' <тип> ';' ;
```

где `<тип>` — это одна из строк `boolean`, `char`, `double`, `integer`; `<идентификатор>` — строка, состоящая из букв, цифр или символов `'_'` и начинающаяся с буквы или символа `'_'`.

Строки `<тип>` и `<идентификатор>`, а также символы `':'` и `','` и `','` являются неделимыми элементами, между которыми (а также до и после них) может находиться произвольное число пробелов.

Гарантируется, что во входном файле количество строк — не более 100, в каждой строке не более 20 переменных, и название каждой переменной не длинее 1000 символов.

Формат выходных данных

В выходной файл необходимо вывести четыре строки, в следующем формате:

```
boolean: <количество объявленных переменных типа boolean>
```

```
char: <количество объявленных переменных типа char>
```

```
double: <количество объявленных переменных типа double>
```

```
integer: <количество объявленных переменных типа integer>
```

Обратите внимание, что между двоеточием и числом переменных должен быть выведен один пробел.

Пример

ВВОД	ВЫВОД
<pre>i, j, k : integer ; flag: boolean; length: integer; c1, c2: char; c3: char;</pre>	<pre>boolean: 1 char: 3 double: 0 integer: 4</pre>

Задача F. Футбольные парадоксы

Ограничение по времени: 2 секунды

Ограничение по памяти: 64 Мб

После того, как сборная Бразилии стала Чемпионом Мира по футболу, она проиграла сборной Италии, которая, в свою очередь, проиграла сборной Болгарии, а та — сборной Люксембурга. Так что же, получается, что Люксембург сильнее Бразилии? Такие парадоксы происходят довольно часто, и удивляться этому не стоит, так как победа в матче не обладает свойством транзитивности, то есть описанное выше не означает, что при встрече сборных Бразилии и Люксембурга обязательно выиграет Люксембург.

Это заинтересовало Петю, и он решил проанализировать результаты всех матчей, которые он знает, и выбрать из них самую длинную цепочку игр, обладающую следующим свойством — победитель любого матча этой цепочки (кроме последнего) должен проиграть в следующем матче этой цепочки. Заметьте, что хронологический порядок игр должен сохраниться, то есть следующий матч в цепочке должен быть сыгран позже предыдущего. Пете не важно, заканчивается ли цепочка игр той же командой, с которой начинается, или нет. Прежде всего, его волнует количество игр в ней.

Формат входных данных

Во входном файле содержится отсортированная хронологически последовательность игр, то есть каждая следующая игра в этой последовательности была сыграна позже предыдущей. В первой строке входного файла записано целое число n — количество сыгранных игр ($0 < n \leq 10000$). В каждой из следующих n строк содержится описание одной игры. Каждая игра описывается строкой из семи символов. Первые три символа — идентификатор выигравшей команды, четвертый символ — тире, символы с пятого по седьмой — идентификатор проигравшей команды. Идентификатор команды всегда является трехбуквенным и состоит только из заглавных латинских букв. Количество различных идентификаторов команд во входном файле не превышает 200.

Формат выходных данных

В первую строку выходного файла выведите максимальное количество матчей в искомой цепочке. Во вторую строку выведите цепочку команд, начиная с команды, победившей в последнем матче цепочки, и заканчивая командой, проигравшей в первом. Если вариантов наиболее длинных цепочек несколько, выведите любой из них.

Примеры

ВВОД	ВЫВОД
5 FRA-ITA GER-ITA ITA-GER ITA-LUX RUS-ITA	3 RUS-ITA-GER-ITA
3 BLR-UKR LAT-BLR UKR-LAT	3 UKR-LAT-BLR-UKR

Пояснение: оптимальная цепочка в первом примере состоит из матчей GER-ITA, ITA-GER, RUS-ITA, а во втором примере все матчи входят в цепочку.