

## Задача А. Пузырьки 1D

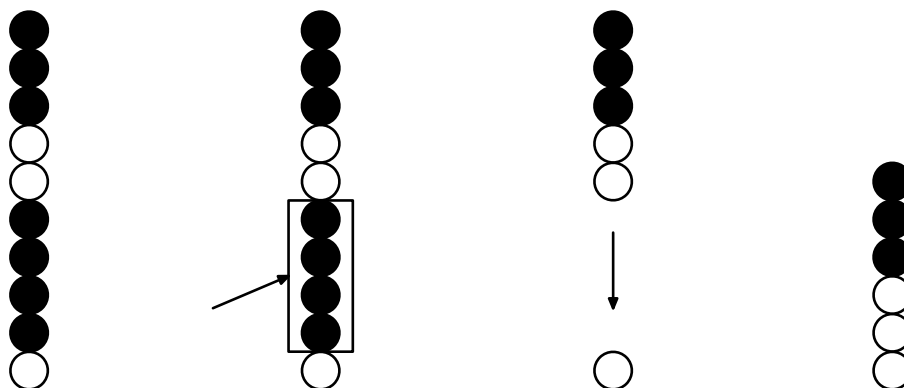
Имя входного файла: `bubbles.in`  
Имя выходного файла: `bubbles.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Сергея — большой любитель игр на сотовом телефоне. Недавно он скачал из интернета новую игру «Пузырьки 1D». Опишем правила игры.

Исходная позиция в игре представляет собой  $n$  пузырьков, расположенных вертикально в ряд. Каждый пузырек окрашен в один из четырех цветов — красный, зеленый, синий или желтый. Назовем *группой* несколько следующих подряд пузырьков одинакового цвета, непосредственно сверху и снизу от которых находятся либо пузырьки другого цвета, либо границы ряда пузырьков.

За один ход разрешается выбрать любую группу, состоящую хотя бы из двух пузырьков, и взорвать ее. За взрыв группы, содержащей  $k$  пузырьков, игрок получает  $k^2$  очков. После взрыва группы пузырьки, которые находились сверху, опускаются вниз.

Например, ниже на рисунке показана позиция, содержащая 10 пузырьков. В ней четыре группы, содержащие 3, 2, 4 и 1 пузырек, соответственно. Если взорвать группу, содержащую четыре пузырька, то игрок получит 16 очков, и верхние 5 пузырьков опустятся вниз. В получившейся позиции 6 пузырьков, и две группы по 3 пузырька в каждой.



По заданной начальной позиции в игре выясните, сможет ли Сергей уничтожить все пузырьки, и если да, то какое максимальное количество очков он сможет заработать.

### Формат входного файла

Входной файл содержит одну строку, состоящую из букв «R», «G», «B» и «Y», описывающую начальную позицию. Буквы задают цвета пузырьков в порядке просмотра сверху вниз («R» означает красный пузырек, «G» — зеленый, «B» — синий, а «Y» — желтый).

В заданной позиции не менее двух и не более 100 пузырьков.

### Формат выходного файла

Выведите в выходной файл одно число — максимальное количество очков, которое сможет заработать Сергей. Если уничтожить все пузырьки невозможно, выведите 0.

### Примеры

<code>bubbles.in</code>	<code>bubbles.out</code>
<code>RRRGRRRRG</code>	34
<code>RB</code>	0

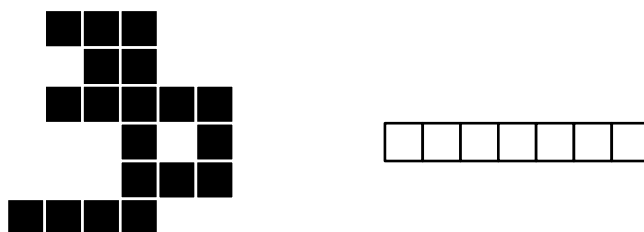
В первом примере следует действовать следующим образом: сначала надо взорвать группу из четырех красных пузырьков, получив 16 очков. Затем надо взорвать в любом порядке получившиеся две группы по 3 пузырька, получив по 9 очков за каждую.

## Задача В. Клетка для хомячка

Имя входного файла: `cage.in`  
Имя выходного файла: `cage.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Андрюше на день рождения подарили хомячка. Пока Андрюша не купил для него клетку, он решил сделать ему клетку из подручных средств. Для изготовления клетки он решил использовать набор кубиков, подаренный ему на прошлый день рождения. Однако, неожиданно выяснилось, что сестра Андрюши склеила кубики суперклеем, и отделить их друг от друга не представляется возможным.

Все кубики оказались склеены в две фигуры. Любые два кубика в каждой из фигур либо не имеют общих точек, либо имеют общую грань, либо имеют общее ребро, но в последнем случае есть кубик, с которым каждый из них имеет общую грань. Каждую фигуру можно положить на стол так, что каждый кубик будет касаться стола одной из своих граней.

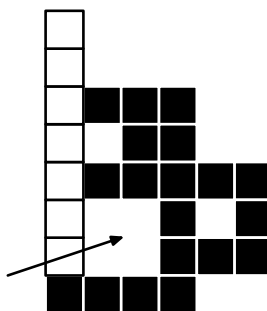


Теперь Андрюша хочет положить эти две фигуры на стол так, чтобы получилась клетка для хомячка. Фигуры должны быть положены таким образом, чтобы каждый кубик касался стола гранью. Стороны нижних граней кубиков должны быть параллельны сторонам стола. Любые два кубика, принадлежащие различным фигурам, должны либо не касаться друг друга, либо иметь общую грань, либо иметь общее ребро. Фигуры разрешается поворачивать и переворачивать.

Положив фигуры, Андрюша собирается выпустить хомячка на стол. Чтобы он не упал со стола, у него не должно быть возможности добраться от точки, в которую Андрюша его выпустит, до края стола. Хомячок не может перелезть через кубики, и, в частности, не может пролезть между двумя кубиками, имеющими общее ребро. Стол существенно больше каждой из фигур.

Андрюша хочет, чтобы площадь, по которой может бегать хомячок, была как можно больше. Помогите ему выяснить, какая максимальная площадь может быть у территории, до которой сможет добраться хомячок. Площадь грани кубика будем считать равной единице.

Например, две фигуры, показанные на рисунке выше, можно расположить как показано на следующем рисунке. Если выпустить хомячка в точку, отмеченную стрелкой, то доступная ему территория будет иметь площадь равную четырём.



### Формат входного файла

Первая строка входного файла содержит два числа:  $h_1$  и  $w_1$  ( $1 \leq h_1, w_1 \leq 10$ ). Следующие  $h_1$  строк содержат по  $w_1$  символов и описывают первую фигуру, вид сверху. Каждый из этих сим-

волов — либо «\*» (звездочка), либо «.» (точка), звездочка обозначает кубик, а точка — пустое место.

Следующая строка содержит два числа:  $h_2$  и  $w_2$  ( $1 \leq h_2, w_2 \leq 10$ ). Следующие  $h_2$  строк содержат по  $w_2$  символов и описывают вторую фигуру в формате, аналогичном первой.

Каждая из фигур связана и содержит хотя бы один кубик.

## Формат выходного файла

Выведите в выходной файл одно число — максимальную площадь, которая может быть доступна хомячку. Если сделать клетку для хомячка невозможно, выведите 0.

## Примеры

cage.in	cage.out
<pre>8 8 ..... .***.... ..**.... .*****. ...*.*.. ...***.. ****.... ..... 8 8 ..... ..... ..... ..... *****. ..... ..... .....</pre>	<pre>4</pre>

## Задача С. Выпуклая оболочка

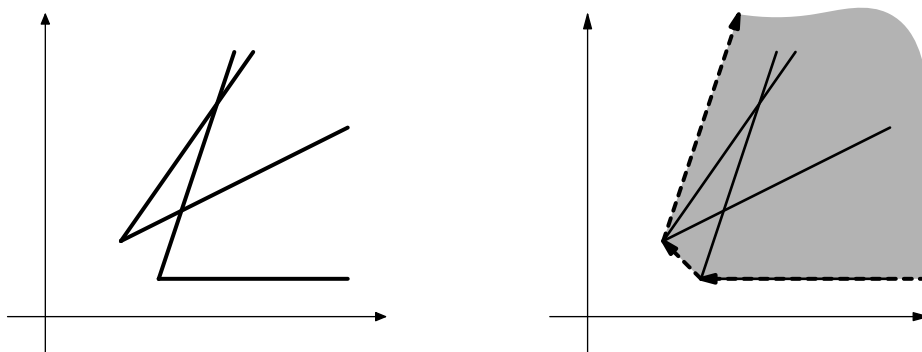
Имя входного файла: `convex.in`  
Имя выходного файла: `convex.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Множество на плоскости называется выпуклым, если вместе с любыми двумя точками оно содержит также и отрезок, соединяющий эти точки. Минимальное по включению множество, содержащее заданное множество точек  $X$ , называется *выпуклой оболочкой* множества  $X$ .

В этой задаче вам требуется найти выпуклую оболочку множества точек, принадлежащих заданному набору углов.

Углом называется геометрическая фигура, образованная двумя лучами, выходящими из одной точки. Эта точка называется вершиной угла.

На рисунке слева приведены два угла, на рисунке справа изображена их выпуклая оболочка.



### Формат входного файла

Первая строка входного файла содержит число  $n$  — количество углов ( $1 \leq n \leq 1000$ ). Каждая из следующих  $n$  строк описывает углы. Каждый угол описывается координатами трех точек: вершины и двух отличных от вершины точек — по одной на каждом из лучей. Все координаты целые и не превышают  $10^4$  по абсолютной величине. Величина угла находится в диапазоне от 0 до 180 градусов, не включительно.

### Формат выходного файла

Выведите в выходной файл границу выпуклой оболочки в виде последовательности направленных лучей, прямых и отрезков. Никакие два объекта в выходном файле не должны лежать на одной прямой. Все отрезки должны иметь длину больше нуля. Объекты должны быть перечислены в таком порядке, чтобы начало каждого следующего совпадало с концом предыдущего. Все числа в выходном файле должны быть целыми и не превосходить  $10^5$  по абсолютной величине. При проходе вдоль описанной границы выпуклая оболочка углов должна быть справа.

На первой строке выведите  $l$  — количество объектов в ответе. Следующие  $l$  строк должны содержать описание объектов. Объекты описываются следующим образом:

- Отрезок: «`segment  $x_1$   $y_1$   $x_2$   $y_2$` », где  $(x_1, y_1)$  и  $(x_2, y_2)$  — концы отрезка, отрезок считается направленным от  $(x_1, y_1)$  к  $(x_2, y_2)$ .
- Луч, направленный от начала: «`outray  $x_1$   $y_1$   $x_2$   $y_2$` », где  $(x_1, y_1)$  — начало луча, а  $(x_2, y_2)$  — произвольная точка на луче, отличная от начала.
- Луч, направленный к началу: «`inray  $x_1$   $y_1$   $x_2$   $y_2$` », где  $(x_2, y_2)$  — начало луча, а  $(x_1, y_1)$  — произвольная точка на луче, отличная от начала.
- Прямая: «`line  $x_1$   $y_1$   $x_2$   $y_2$` », где  $(x_1, y_1)$  и  $(x_2, y_2)$  — две точки на прямой, причем при движении вдоль прямой в ее направлении точка  $(x_1, y_1)$  следует ранее точки  $(x_2, y_2)$ .

Если выпуклой оболочкой является вся плоскость, то выведите  $l = 0$ .

## Примеры

convex.in	convex.out
2 3 1 4 1 4 4 2 2 4 3 3 4	3 inray 4 1 3 1 segment 3 1 2 2 outray 2 2 3 5
2 0 0 1 0 0 1 0 0 -1 0 0 1	1 line 1 0 -1 0
2 0 0 1 0 0 1 0 0 -1 0 0 -1	0

## Задача D. Выражение

Имя входного файла: `expr.in`  
Имя выходного файла: `expr.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Петя — большой любитель математических головоломок. Недавно он прочитал в одном популярном журнале о новой головоломке. Он пытался ее решить несколько дней, но это ему так и не удалось. Помогите Пете справиться с неподдающейся задачей.

В ряд выписаны  $n$  чисел. Требуется поставить между каждой парой соседних чисел один из знаков «+» или «×» таким образом, чтобы значение получившегося выражения было как можно больше. Использовать скобки не разрешается.

Например, для последовательности чисел 1, 2, 3, 1, 2, 3 оптимально расставить знаки следующим образом:  $1 + 2 \times 3 \times 1 \times 2 \times 3$ . Значение выражения в этом случае равно 37.

### Формат входного файла

Первая строка входного файла содержит число  $n$  ( $2 \leq n \leq 200\,000$ ). Вторая строка содержит  $n$  целых чисел — числа, между которыми следует расставить знаки. Все числа находятся в диапазоне от 0 до  $10^9$ .

### Формат выходного файла

Выведите в выходной файл оптимальное выражение. В качестве знака «×» выводите символ «\*» (звездочку). Если оптимальных решений несколько, выведите любое из них.

### Примеры

<code>expr.in</code>	<code>expr.out</code>
6 1 2 3 1 2 3	1+2*3*1*2*3

## Задача Е. Справедливая последовательность

Имя входного файла: `fair.in`  
Имя выходного файла: `fair.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Последовательность из нулей и единиц четной длины назовем *справедливой*, если на четных местах этой последовательности столько же единиц, сколько на нечетных. Например, последовательность «011011» является справедливой, а последовательность «011101» — нет.

Задана некоторая последовательность нечетной длины из нулей и единиц. Из нее разрешается удалить одну цифру. Какую цифру следует удалить, чтобы последовательность стала справедливой?

Например, из последовательности «0111011» с этой целью можно удалить вторую цифру.

### Формат входного файла

Входной файл содержит одну строку. Эта строка содержит последовательность нечетной длины из нулей и единиц. Длина последовательности не превышает 200 001.

### Формат выходного файла

Выведите в выходной файл одно число — номер цифры в последовательности, которую следует удалить, чтобы последовательность стала справедливой. Цифры нумеруются, начиная с 1.

Если это сделать невозможно, выведите 0.

Если решений несколько, выведите любое.

### Примеры

<code>fair.in</code>	<code>fair.out</code>
0111011	2

## Задача F. Развлечения с измерителем

Имя входного файла: `fun.in`  
Имя выходного файла: `fun.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Дима обнаружил у папы на столе специальный чертежный прибор, похожий на циркуль — измеритель. Измеритель отличается от обычного циркуля тем, что в обеих его ножках находятся иголки (у обычного циркуля в одной ножке находится иголка, а в другой — грифель).

Дима взял клетчатый лист бумаги, установил между иглами измерителя некоторое расстояние, прочно зафиксировав его, и начал втыкать измеритель в лист бумаги. Каждый раз Дима втыкал в лист обе иглы измерителя, при этом он всегда делал это так, что дырочки получались в точках пересечений линий, которыми лист разлинован на клетки. При этом в одну и ту же дырку Дима мог вставлять измеритель несколько раз.

Вечером папа нашел лист, с которым развлекался Дима, и решил выяснить, какое расстояние между иглами измерителя Дима мог установить. Все, что знает папа — координаты дырок, проделанных иглами измерителя. Помогите Папе решить поставленную задачу.

### Формат входного файла

Первая строка входного файла содержит число  $n$  — количество дырок ( $2 \leq n \leq 1000$ ). Следующие  $n$  строк содержат по два целых числа — координаты дырок. Координаты не превышают  $10^4$  по абсолютной величине.

### Формат выходного файла

На первой строке выходного файла выведите  $k$  — количество различных расстояний, которые Дима мог установить между иглами измерителя. Следующие  $k$  строк должны содержать по одному вещественному числу — искомые расстояния. Расстояния должны быть выведены в возрастающем порядке. Каждое число должно быть выведено с точностью не менее чем  $10^{-9}$ .

Гарантируется, что существует по крайней мере одно расстояние, которое Дима мог установить между иглами измерителя.

### Примеры

<code>fun.in</code>	<code>fun.out</code>
4	2
0 0	1.0
1 1	1.414213562373
1 0	
0 1	



## Задача G. Жребий Крижановского

Имя входного файла: `game.in`  
Имя выходного файла: `game.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Петя играет с друзьями в игру, которую иногда называют «Жребий Крижановского». Правила игры следующие: в каждом туре каждый игрок загадывает произвольное натуральное число. После этого игрок, загадавший минимальное число, которое не повторяется, выигрывает в этом туре, причем его выигрыш равен этому числу. Например, если играют 6 человек и были загаданы числа 3, 2, 1, 1, 4 и 2, то выиграл первый игрок, причем его выигрыш равен 3. Если все загаданные числа повторяются, то тур считается ничейным и никто баллов не получает.

Общий выигрыш игрока за игру равен сумме баллов за все сыгранные туры.

Петя с друзьями при игре просто называют по очереди загаданные ими числа, а потом определяют, кто выиграл, и подсчитывают баллы. Однако при таком формате игры в принципе можно жульничать, не загадывая число заранее, а, уже зная числа, названные предыдущими игроками, выбрать себе оптимальное «загаданное» число. Этим и пользуется Петя. Он называет число последним и старается выбрать число так, чтобы максимизировать свой выигрыш.

Идет последний тур игры. Известны очки всех игроков перед этим туром и названные игроками числа. Выясните, какое число следует назвать Пете, чтобы по результатам игры у как можно большего числа игроков количество баллов было меньше чем у него. Если таких чисел несколько, то Петя хочет назвать минимальное возможное.

### Формат входного файла

Первая строка входного файла содержит число  $n$  — количество игроков ( $2 \leq n \leq 100$ ). Вторая строка содержит  $n$  чисел — баллы игроков перед последним туром (неотрицательные целые числа, не большие 100). Баллы перечислены в том порядке, в котором игроки обычно называют числа (то есть Петины баллы указаны последними). Третья строка содержит  $n - 1$  число — числа, названные игроками в последнем туре (числа не превышают 100), в том порядке, в котором они их называли.

### Формат выходного файла

Выведите в выходной файл число, которое следует назвать Пете.

### Примеры

<code>game.in</code>	<code>game.out</code>
6 0 0 0 0 0 0 2 3 4 5 6	1
6 8 3 12 5 0 9 2 1 3 1 4	2

Во втором примере Петя не может выиграть в последнем туре. Однако, назвав число 2, Петя не позволяет выиграть первому игроку, и тем самым остается вторым по итогам всей игры. У четырех игроков баллы меньше чем у Пети.

## Задача Н. Программа психологического исследования программистов

Имя входного файла: `program.in`  
Имя выходного файла: `program.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Компания «Макрохард» заказала у одного известного психолога полное психологическое обследование всех работников компании. Психолог, привлеченный для проведения обследования, известен своим инновационным методом, позволяющим составить полную психологическую картину сотрудника по наиболее часто используемому им в программах идентификатору. Однако, к сожалению, программа, используемая в анализе, оказалась неожиданно испорчена вирусом, поэтому требуется срочно написать новую. Помогите известному психологу. Напишите программу, которая по приведенной программе выяснит наиболее часто используемый в ней идентификатор.

Поскольку разные сотрудники компании пишут программы на разных языках программирования, ваша программа должна уметь работать с произвольным языком. Поскольку в разных языках используются различные ключевые слова, то список ключевых слов в анализируемом языке предоставляется на вход программе. Все последовательности из латинских букв, цифр и знаков подчеркивания, которые не являются ключевыми словами и содержат хотя бы один символ, не являющийся цифрой, могут быть идентификаторами. При этом в некоторых языках идентификаторы могут начинаться с цифры, а в некоторых — нет. Если идентификатор не может начинаться с цифры, то последовательность, начинающаяся с цифры, идентификатором не является. Кроме этого, задано, является ли язык чувствительным к регистру символов, используемых в идентификаторах и ключевых словах.

### Формат входного файла

Первая строка входного файла содержит число  $n$  — количество ключевых слов в языке ( $0 \leq n \leq 50$ ), и два слова  $s$  и  $d$ , каждое из которых равно либо «yes», либо «no». Слово  $s$  равно «yes», если идентификаторы и ключевые слова в языке чувствительны к регистру символов, и «no», если нет. Слово  $d$  равно «yes», если идентификаторы в языке могут начинаться с цифры, и «no», если нет.

Следующие  $n$  строк содержат по одному слову, состоящему из букв латинского алфавита и символов подчеркивания — ключевые слова. Все ключевые слова непусты, различны, при этом, если язык не чувствителен к регистру, то различны и без учета регистра. Длина каждого ключевого слова не превышает 50 символов.

Далее до конца файла идет текст программы. Он содержит только символы с ASCII-кодами от 32 до 126 и переводы строки. В программе есть хотя бы один идентификатор.

### Формат выходного файла

Выведите в выходной файл идентификатор, встречающийся в программе максимальное число раз. Если таких идентификаторов несколько, следует вывести тот, который встречается в первый раз раньше. Если язык во входном файле не чувствителен к регистру, то можно выводить идентификатор в любом регистре.

### Примеры

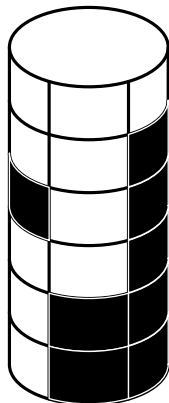
program.in	program.out
<pre>0 yes no int main() {     int a;     int b;     scanf("%d%d", &amp;a, &amp;b);     printf("%d", a + b); }</pre>	<pre>int</pre>

program.in	program.out
<pre>0 yes no #define INT int int main() {     INT a, b;     scanf("%d%d", &amp;a, &amp;b);     printf("%d %d", a + b, 0); }</pre>	d
<pre>6 no no program var begin end while for program sum; var     A, B: integer; begin     read(A, b);     writeln(a + b); end.</pre>	a
<pre>1 yes yes - a = 0h b = 0h c = 0h</pre>	0h

## Задача I. Головоломка

Имя входного файла: puzzle.in  
Имя выходного файла: puzzle.out  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Пете на день рождения подарили новую головоломку. Головоломка представляет собой цилиндр, состоящий из  $n$  круглых слоев, нанизанных на одну вертикальную ось. Каждый слой можно вращать независимо от других. Каждый слой разбит на  $n$  квадратиков, каждый из которых может быть либо черным, либо белым. В устойчивом состоянии квадратики соседних слоев находятся в точности друг под другом.



Для задания конфигурации головоломки удобно рассмотреть ее развертку — «разрезать» поверхность цилиндра вдоль вертикальной линии, проходящей по границам квадратиков, и обозначить черные клетки символом «0», а белые — символом «1». Пусть, например, одна из возможных разверток головоломки, приведенной на рисунке, следующая (на рисунке видно только первые три столбца этой развертки):

```
000110
001110
101000
001000
011111
011110
```

Задача решающего головоломку состоит в том, чтобы, поворачивая слои, добиться того, чтобы все вертикальные столбцы были различны. Например, головоломка приведенная выше, не решена, поскольку два из ее столбцов (четвертый и пятый на приведенной развертке) одинаковы. Если же повернуть нижний слой влево на один квадратик, развертка головоломки примет следующий вид:

```
000110
001110
101000
001000
011111
111100
```

Теперь все столбцы различны и следовательно головоломка решена.

Для того, чтобы решать головоломку было интереснее, на ее раскраску наложено дополнительное условие: нельзя повернуть один из слоев головоломки меньше чем на полный оборот таким

образом, что внешний вид головоломки останется тем же. Так, например, для  $n = 6$  слой с раскраской «010101» не разрешается, поскольку при его повороте на 2 квадратика внешний вид головоломки не меняется.

По заданной развертке головоломки выясните, можно ли ее решить, и если да, то приведите пример развертки решенной головоломки.

### Формат входного файла

Первая строка входного файла содержит число  $n$  — количество слоев в головоломке и количество квадратиков в одном слое ( $1 \leq n \leq 200$ ). Следующие  $n$  строк содержат по  $n$  символов, каждый из которых равен 0 или 1 — развертку головоломки.

### Формат выходного файла

Если решить головоломку можно, на первой строке выходного файла выведите слово «Yes». В этом случае следующие  $n$  строк должны содержать произвольную развертку решенной головоломки.

Если решить головоломку нельзя, выведите «No» на первой строке выходного файла.

### Примеры

puzzle.in	puzzle.out
6	Yes
000110	000110
001110	001110
101000	101000
001000	001000
011111	011111
011110	111100

## Задача J. Гражданская оборона

Имя входного файла: `shelter.in`  
Имя выходного файла: `shelter.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Штаб гражданской обороны Тридесятой области решил обновить план спасения на случай ядерной атаки. Известно, что все  $n$  селений Тридесятой области находятся вдоль одной прямой дороги. Вдоль дороги также расположены  $m$  бомбоубежищ, в которых жители селений могут укрыться на случай ядерной атаки.

Чтобы спасение в случае ядерной тревоги проходило как можно эффективнее, необходимо для каждого селения определить ближайшее к нему бомбоубежище.

### Формат входного файла

Первая строка входного файла содержит число  $n$  — количество селений ( $1 \leq n \leq 100\,000$ ). Вторая строка содержит  $n$  различных целых чисел,  $i$ -е из этих чисел задает расстояние от начала дороги до  $i$ -го селения.

Третья строка входного файла содержит число  $m$  — количество бомбоубежищ ( $1 \leq m \leq 100\,000$ ). Четвертая строка содержит  $m$  различных целых чисел,  $i$ -е из этих чисел задает расстояние от начала дороги до  $i$ -го бомбоубежища.

Все расстояния положительны и не превышают  $10^9$ . Селение и убежище могут располагаться в одной точке.

### Формат выходного файла

Выведите в выходной файл  $n$  чисел — для каждого селения выведите номер ближайшего к нему бомбоубежища. Бомбоубежища пронумерованы от 1 до  $m$  в том порядке, в котором они заданы во входном файле.

### Примеры

<code>shelter.in</code>	<code>shelter.out</code>
4	2 2 1 1
1 2 6 10	
2	
7 3	

## Задача К. Движение по полосам

Имя входного файла: `traffic.in`  
Имя выходного файла: `traffic.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

При организации движения по сложным перекресткам, для того, чтобы траектории водителей, выполняющих различные маневры не пересекались, вводят ограничения на возможные маневры водителей, в зависимости от того, по какой полосе движения водитель подъехал к перекрестку. Для этого используется знак «движение по полосам», на рисунке справа приведен пример такого знака, установленного перед одним из перекрестков в Санкт-Петербурге.



Рассмотрим дорогу, подходящую к перекрестку, на котором сходится  $m$  дорог. Водитель, подъезжающий к перекрестку по этой дороге, потенциально может продолжить свое движение в  $m$  различных направлениях — обратно по дороге, по которой он приехал, а также по одной из оставшихся  $m - 1$  дорог. Пронумеруем возможные направления числами от 1 до  $m$  слева направо с точки зрения подъезжающего водителя, номер 1 получит разворот и возврат по дороге, по которой водитель подъезжал к перекрестку, номер 2 — поворот на самую левую из дорог, и т. д.

Пусть дорога содержит  $n$  полос для движения. Пронумеруем полосы от 1 до  $n$  слева направо, самая левая полоса получит номер 1, следующая номер 2, и т. д. Знак «движение по полосам» разрешает каждой из полос движение по некоторым из  $m$  возможных направлений. При этом должны выполняться следующие условия:

1. если с  $i$ -й полосы разрешено движение в  $a$ -м направлении, а с  $j$ -й полосы — в  $b$ -м направлении, причем  $i < j$ , то  $a \leq b$ ;
2. с каждой полосы разрешено движение хотя бы в одном направлении;
3. в каждом направлении разрешено движение хотя бы с одной полосы.

Инспекция по безопасности дорожного движения заинтересовалась, а сколько различных знаков «движение по полосам» можно установить перед таким перекрестком. Помогите им найти ответ на этот вопрос.

### Формат входного файла

Входной файл содержит два целых числа:  $m$  и  $n$  ( $2 \leq m \leq 50$ ,  $1 \leq n \leq 15$ ).

### Формат выходного файла

В выходной файл выведите одно число — количество возможных знаков «движение по полосам», которые можно установить перед перекрестком.

### Примеры

<code>traffic.in</code>	<code>traffic.out</code>
4 2	7

В примере возможны следующие варианты знаков «движение по полосам»:

С левой полосы	С правой полосы
разворот	разворот, налево, прямо, направо
разворот	налево, прямо, направо
разворот, налево	налево, прямо, направо
разворот, налево	прямо, направо
разворот, налево, прямо	прямо, направо
разворот, налево, прямо	направо
разворот, налево, прямо, направо	направо