Задача A. Скобки

Имя входного файла:	`brackets.in`
Имя выходного файла:	`brackets.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Рассмотрим скобочные последовательности с одним типом скобок. Для заданной скобочной последовательности найдите количество ее подпоследовательностей, которые являются правильными скобочными последовательностями.

Например, для последовательности “`((())())(`” таких последовательностей восемь: “`((())())`”, “`(())()`”, “`((()))`”, “`()())`”, “`(())`”, “`()()`”, “`()`” и “”.

Формат входного файла

Входной файл содержит последовательность, состоящую не более чем из 300 круглых скобок.

Формат выходного файла

Выведите количество различных правильных скобочных подпоследовательностей заданной последовательности.

Примеры

brackets.in	**brackets.out**
`((())())(`	`8`

Задача B. Карточки

Имя входного файла:	`cards.in`
Имя выходного файла:	`cards.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Вася и Петя играют в следующую игру. Вася кладет на стол два ряда карточек. Первый ряд состоит из n карточек, на каждой из которых написано некоторое число a_i. Второй ряд состоит из n карточек, на каждой из которых написано некоторое число b_i.

Пете требуется переупорядочить карточки первого ряда так, чтобы на столе получилось два одинаковых ряда карточек. За одну секунду Петя может поменять местами i-ую и $i+1$-ую $(1 \le i \le n-1)$ карточки первого ряда.

Помогите Пете переупорядочить карточки затратив на это минимальное время.

Формат входного файла

Первая строка входного файла содержит целое число n $(1 \le n \le 100\,000)$. Вторая строка содержит n целых чисел a_i. Третья строка содержит n целых чисел b_i. Все числа по абсолютной величине не превосходят 10^6.

Формат выходного файла

В первой строке выходного файла выведите количество секунд, за которые Петя сможет переупорядочить карточки требуемым образом. Если переупорядочить карточки требуемым образом невозможно, выведите единственное число -1.

Примеры

cards.in	cards.out
5 3 2 3 4 5 5 4 3 2 3	7
8 1 2 1 2 1 2 1 2 2 1 2 1 2 1 2 2	-1

Задача C. Список

Имя входного файла:	`list.in`
Имя выходного файла:	`list.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В наше время создатели офисных приложений стараются сделать все для удобства пользователя. Поэтому даже такая мелочь, как представление на экране списков чисел — например, для вывода номеров страниц, — должна быть тщательно проработана.

Вы должны реализовать функцию, которая по заданному набору целых чисел будет формировать строку, являющуюся его самым коротким текстовым представлением. Текстовое представление — строка, состоящая из разделенных запятыми чисел и диапазонов чисел вида «a`, ..., `b», которые используются для записи набора всех чисел от a до b. При этом все числа, входящие в строку должны быть упорядочены по возрастанию в том порядке, в котором они встречаются в строке.

Формат входного файла

В первой строке входного файла содержится целое число n ($1 \le n \le 1000$) — размер набора. Вторая строка содержит n задающих набор целых чисел, по абсолютной величине не превосходящих 10000, разделенные пробелами. Одно число может встречаться в этом описании несколько раз.

Формат выходного файла

В первой строке выходного файла запишите одно из кратчайших текстовых представлений заданного набора чисел. Следите за правильной расстановкой пробелов. Выходной файл в примере содержит ровно три пробела.

Примеры

list.in	list.out
7 1 3 5 -1 3 4 6	-1, 1, 3, ..., 6

Задача D. Новый год

Имя входного файла:	`newyear.in`
Имя выходного файла:	`newyear.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Санта Клаус решил сделать новогодний подарок своим эльфам. Конечно же, Санта был бы не Санта, если бы его подарок был обыкновенным. Подарок просто замечательный — эльфы идут в кино!

Конечно же, если все эльфы пойдут в кино одновременно, это может показаться кому-нибудь подозрительным. Ну, вы понимаете, некоторые люди все еще не верят в эльфов. Так что каждый день ровно одна пара эльфов пойдет в кино. Конечно, идти в кино самому по себе неинтересно, гораздо интереснее пойти с другом или подругой. Таким образом, каждый день один мальчик-эльф и одна девочка-эльф вместе пойдут в кино.

Но эльфы, они такие разные. Каждый эльф четко знает, с какими эльфам и на какой фильм он или она хотели бы пойти. Поскольку у каждой пары эльфов свои взаимоотношения, для каждой пары, которая согласна вместе пойти в кинотеатр, известно какой именно фильм они хотели бы посмотреть.

Но на самом деле все еще хуже. Дело в том, что билеты в кино стоят денег. Конечно, Санта не беден, но ему нужно много денег на подарки детям. Так что собрав у эльфов пожелания, кто куда и с кем хотел бы пойти в кино, Санта решил удовлетворить некоторые пожелания таким образом, чтобы потратив как можно меньше денег, добиться того, что каждый эльф хотя бы один раз побывает в кино.

Помогите Санте, он пока не до конца освоился с компьютером.

Формат входного файла

Первая строка входного файла содержит два целых числа n и m — количество мальчиков-эльфов и девочек-эльфов, соответственно ($1 \le n, m \le 100$). Вторая строка содержит r — количество пожеланий, которое Санта получил от своих эльфов ($1 \le r \le 1000$). Следующие r строк содержат по три целых числа a_i, b_i и c_i каждая. Числа означают, что мальчик-эльф a_i хотел бы пойти в кино с девочкой-эльфом b_i, и билет на фильм, на который они хотели бы пойти, стоит c_i ($1 \le c_i \le 1000$). Каждый эльф хочет пойти в кино хотбя бы с одним другим эльфом.

Формат выходного файла

На первой строке выходного файла выведите минимальную сумму, которую Санте придется потратить, чтобы сделать своим эльфам подарок. На второй строке выведите k — количество билетов, которое Санта должен купить. Наконец, третья строка должна содержать k целых чисел — номера пожеланий, которые следует удовлетворить.

Примеры

newyear.in	newyear.out
3 3 7 1 1 3 1 2 2 1 3 4 2 1 3 2 2 9 3 1 2 3 3 11	11 4 2 3 4 6

Задача E. Отношения

Имя входного файла:	`relations.in`
Имя выходного файла:	`relations.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Бинарным отношением R на множестве X называется множество упорядоченных пар элементов из X: $R \subset X \times X$. Если X конечно и содержит n элементов, то отношение можно задать как квадратную булеву матрицу размера $n \times n$.

В некоторых случаях бывает важно задать отношение в более компактной форме. Один из способов компактного задания отношений применяется, в частности, при описании грамматик операторного предшествования.

Рассмотрим две функции f и g, каждая из которых сопоставляет элементам X целые числа. Будем говорить, что эти функции описывают отношение R, если для любых x и y из X условие $(x, y) \in R$ выполнено тогда и только тогда, когда $f(x) \le g(y)$.

По заданному отношению R, найдите способ описать его указанным образом с помощью двух функций f и g, либо выясните, что это невозможно сделать.

Формат входного файла

Первая строка входного файла содержит n — количество элементов в множестве $X = \{x_1, x_2, \ldots x_n\}$ ($1 \le n \le 1\,000$). Следующие n строк описывают отношение R. Каждая строка содержит n символов, j-й символ i-й из этих строк равен '1', если $(x_i, x_j) \in R$, и '0' в противном случае.

Формат выходного файла

Если отношение можно описать указанным образом, выведите "YES" на первой строке выходного файла. В этом случае вторая строка должна содержать n целых чисел в диапазоне от -10^9 до 10^9 — значения функции f на элементах x_1, x_2, $\ldots$, x_n, соответственно, а третья строка должна описывать функцию g аналогичным образом.

Если отношение нельзя закодировать описанным образом с помощью двух функций, выведите "NO" на первой строке выходного файла.

Примеры

relations.in	relations.out
3 111 110 100	YES 0 1 2 2 1 0
3 110 101 011	NO

Задача F. Снова ладьи

Имя входного файла:	`rooks.in`
Имя выходного файла:	`rooks.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Напомним, что ладья — это шахматная фигура, которая ходит по вертикалям и горизонталям.

Будем называть расстановку k белых и l черных ладей на доске размера $m \times n$ мирной, если ни одна ладья не угрожает ладье другого цвета.

Для заданных m, n, k и l найдите количество мирных расстановок ладей на доске.

Формат входного файла

Входной файл содержит четыре целых числа: m, n, k и l ($2 \le m, n \le 10$, $1 \le k$, $1 \le l$, $k + l \le mn$).

Формат выходного файла

Выведите одно целое число — количество мирных расстановок k белых и l черных ладей на доске размера $m \times n$.

Примеры

rooks.in	rooks.out
3 3 2 2	18

Задача G. Покупки

Имя входного файла:	`shopping.in`
Имя выходного файла:	`shopping.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Во многих фирмах, занимающихся торговлей, существует должность менеджера по закупкам. Как известно, они занимаются тем, что по торговому плану, представляющему собой список наименований товаров, для каждого из которых указано необходимое количество, закупает указанные в нем товары на оптовых базах. Торговый план при этом составляется руководством компании. Иногда у менеджеров по закупкам возникает желание принести выгоду не только своей фирме, но и себе.

Только что, как раз после подписания очередного торгового плана на заказ n наименований товаров, открылась новая оптовая база. Как это часто бывает сразу после открытия, ее цены на многие товары ниже заложенных в план. Наверное, этим можно воспользоваться.

На закупку товаров были выделены деньги из расчета того, что все товары будут закупаться на старой оптовой базе. Менеджер хочет, воспользовавшись возможностью покупать товары на новой базе, потратить как можно меньше денег на закупку требуемого количества товаров (непотраченные деньги он, конечно, сможет забрать себе).

Чтобы не вызывать сильных подозрений, производить на новой базе закупки, суммарная запланированная стоимость которых была больше, чем d денежных единиц, не следует. Осталось только рассчитать, какие товары и в каком количестве следует закупать на новой базе, чтобы осталось как можно больше непотраченных денег.

Формат входного файла

Первая строка содержит четыре числа: n $(1 \le n \le 1000)$, d, а так же k_1 и k_2 $(1 \le k_1, k_2 \le 1000)$ — количества наименований товаров, имеющихся на открытых ранее и новой базе соответственно. После этого идут n строк, каждая из которых содержит название товара в плане и его количество (положительное вещественное число). За ними следуют два блока из k_1 и k_2 строк соответственно, отделенные от предыдущего и разделенные между собой переводом строки — наименования товаров на базах и цены за единицу товара соответственно. Все цены являются положительными числами, даже на новой базе.

Названия товаров состоят из не более, чем 100 латинских букв и символов подчеркивания, при этом регистр букв не учитывается. Вещественные числа заданы не более чем с двумя знаками после десятичной точки и по величине не превосходят 10^6.

Формат выходного файла

В выходной файл выведите n вещественных чисел, по одному на строке, задающих количество соответствующего товара, закупаемого на новой базе. На i-ой строке выведите количество товара, идущего i-ым в плане. Ошибки менее 0.01 будут игнорироваться.

Примеры

shopping.in	shopping.out
4 11.0 5 4 sugar 0.5 powdered_milk 7 flour 8 salt 3 salt 0.5 flour 1 sugar 20 powdered_milk 3 cinnamon 8 sugar 10 flour 2 salt 0.4 liquid_hydrogen 10000	0.5000 0.0000 0.0000 2.0000

Задача H. TeXнические проблемы

Имя входного файла:	`tex.in`
Имя выходного файла:	`tex.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Известно, что многие люди делают мелкие ошибки, набирая документы с помощью системы верстки TeX. Одна из наиобее распространенных ошибок — забытые фигурные скобки вокруг показателя степени или нижнего индекса в том случае, если он представляет собой отрицательное число, или число, состоящее более чем из одной цифры.

Рассмотрим несколько примеров. Чтобы получить в документе “10^{-3}” следует набрать “`$10^{-3}$`”. Если забыть фигурные скобки и набрать вместо этого “`$10^-3$`”, то получится “10^-3” — обычно это совсем не то, чего хочется. Аналогично, чтобы получить “k_{13}” следует набрать “`$k_{13}$`”. Если вместо этого набрать “`$k_13$`”, то получится “k_13”.

Опишем фрагмент синтаксиса TeXа, который требуется в этой задаче. Формулы в TeXе набираются в так называемом математическом режиме. Существует два типа формул — формулы в тексте и «выключенные» формулы. Формулы в тексте располагаются между двумя знаками доллара ‘`$`’, выключенные формулы располагаются между двумя двойными знаками доллара. Заметьте, что в случае двойного знака доллара между ними нет пробелов.

Чтобы набрать собственно знак доллара, его надо защитить. Чтобы защитить символ доллара, надо поставить перед ним символ . ‘`\$`’. Этот символ в свою очередь тоже можно защитить, комбинация ‘`\\`’ имеет специальное значение — перевод строки в документе.

Верхний индекс или показатель степени создается с помощью символа ‘`^`’, нижний индекс — с помощью символа ‘`_`’. По умолчанию каждый из этих символов действует только на символ, который находится непосредственно после него. Чтобы создать верхний или нижний индекс, состоящий из более чем одного символа, его надо взять в фигурные скобки: ‘`{`’ и ‘`}`’.

Каждый из символов ‘`^`’, ‘`_`’, ‘`{`’ и ‘`}`’ в свою очередь может быть защищен обратной косой чертой ‘`\`’. Два или более незащищенных символа ‘`^`’ или ‘`_`’ не встречаются подряд в TeX-е, и не будут встречаться во входных файлах к этой задаче.

Будем считать целым числом произвольную последовательность цифр от ‘`0`’ до ‘`9`’, перед которой может стоять знак минус: ‘`-`’.

Ваша задача — найти все незащищенные символы ‘`^`’ и ‘`_`’ в математическом режиме и выдать предупреждение для каждого случая, когда непосредственно после них идет целое число, состоящее более чем из одного символа.

Формат входного файла

Входной файл содержит некоторый корректный документ TeX. Его длина не превышает 10 килобайт.

Гарантируется, что входной файл корректен в следующем смысле:

- блоки текста, которые находятся в математическом режиме, корректно ограничены символами доллара;
- блоки в математическом режиме не находятся один внутри другого;
- документ не заканчивается в математическом режиме;
- нигде в документе нет более чем двух последовательных символов доллара;
- после незащищенного символа ‘`^`’ или ‘`_`’ не следует пробела, доллара, другого символа ‘`^`’ или ‘`_`’, либо конца файла.

Формат выходного файла

Для каждого незащищенного символа ‘`^`’ или ‘`_`’ во входном файле, который находится в математическом режиме, в случае, если после него следует целое число, состоящее более чем из одного символа, выведите предупреждение. Следуйте формату, приведенному в примере выходного файла.

Позиция, которую следует выводить — это позиция символа ‘^’ или ‘_’, после которого следует число.

Если подозрительных моментов во входном файле нет, выведите “No warnings” в выходной файл.

Примеры

tex.in

```
It is well known that the kinetic energy equals $$E=mv_2 \atop 2.$$

The tenth power may be mistakenly typeset as $10_10$ when
the proper way is $10_{10}$.

The dollar sign can be typeset as \$. In this case
it doesn't turn the math mode on and 10_10 should issue no warning.

However, backslash can also be escaped and the following
dollar is not escaped any more\\$k_-3$

What can we do with \TeX? Essentialy everything. We can display
physics formulas, such as $p_-+n_+=0$.
```

tex.out

```
Warning at line 3, position 49
Warning at line 10, position 35
```

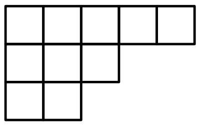
Задача I. Увидеть Юнга и умереть

Имя входного файла:	young.in
Имя выходного файла:	young.out
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

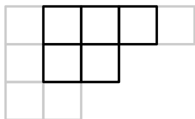
Диаграммы Юнга используются для того, чтобы изобразить разбиение числа на слагаемые. Разбиение числа n на слагаемые представляет собой сумму вида $n = m_1 + m_2 + \ldots + m_k$, где $m_1 \geq m_2 \geq \ldots \geq m_k$.

Диаграмма состоит из n квадратиков, организованных в виде k рядов, где k количество слагаемых в разбиении. Ряд, соответствующий числу m_i, содержит m_i квадратиков. Все ряды выровнены по левому краю и упорядочены от более длинного к более короткому.

Например, диаграмма Юнга, приведенная на рисунке, соответствует разбиению $10 = 5 + 3 + 2$.



Иногда можно *вписать* одну диаграмму Юнга в другую. Диаграмму X можно вписать в диаграмму Y, если можно удалить некоторые квадратики из диаграммы Y так, чтобы получилась диаграмма X. Отметим, что разрешается только удалять некоторые квадратики, вращать или отражать диаграмму не разрешается. Например, диаграмма для разбиения $5 = 3 + 2$ может быть вписана в диаграмму для разбиения $10 = 5 + 3 + 2$, как показано на рисунке.



С другой стороны, диаграмму для разбиения $8 = 4 + 4$ нельзя вписать в диаграмму для разбиения $10 = 5 + 3 + 2$.

Для заданного n найдите такое разбиение n на слагаемые, что в соответствующую ему диаграмму Юнга можно вписать максимальное количество различных диаграмм.

Например, в диаграмму для разбиения $10 = 5 + 3 + 2$ можно вписать 36 различных диаграмм. Однако это не максимальное значение. В диаграмму для разбиения $10 = 4 + 2 + 2 + 1 + 1$ можно вписать 41 диаграмму Юнга.

Формат входного файла

Входной файл содержит целое число n ($1 \leq n \leq 100$).

Формат выходного файла

На первой строке выходного файла выведите максимальное число диаграмм Юнга, которые можно вписать в некоторую диаграмму, соответствующую разбиению на слагаемые числа n.

На второй строке выведите одно или более целых чисел — количество квадратиков в каждом из рядов оптимальной диаграммы.

Примеры

young.in	young.out
10	41 4 2 2 1 1