Задача A. Туризм

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Министерство туризма Лайнландии решило разработать интересный маршрут для туристов. В Лайнландии n городов, расположенных вдоль одной дороги. Города пронумерованы от 1 до n вдоль дороги, причём в некоторых городах есть достопримечательности, а в других — нет. Введем обозначение: пусть $a_i = 1$, если в i-м городе есть достопримечательности, и $a_i = 0$, если нет.

Выбор маршрута будет осуществлять министр туризма. Для того, чтобы ему было из чего выбирать, решено было разработать два различных аналогичных маршрута. Каждый маршрут должен представлять собой несколько идущих подряд вдоль дороги городов: $l, l + 1, \ldots, r$. Два маршрута считаются аналогичными, если они посещают одинаковое количество городов, причём количество городов в маршрутах, содержащих достопримечательности, также совпадает.

Чтобы маршрут был как можно интереснее, он должен посещать максимальное возможное количество городов. Помогите сотрудникам министерства, найдите два аналогичных маршрута, содержащих максимальное возможное количество городов.

Формат входных данных

В единственной строке входных данных расположена строка из символов «0» и «1», i-й символ этой строки задает значение a_i. Длина этой строки хотя бы 3 и не превосходит 10^6. Символы не разделяются пробелами.

Формат выходных данных

Выведите четыре числа s_1, t_1, s_2 и t_2 — номера первого и последнего города первого маршрута и номера первого и последнего города второго маршрута, соответственно. Гарантируется, что ответ всегда существует.

Пример

стандартный ввод	стандартный вывод
10101	1 4 2 5

Задача B. Открытый кубок

Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

К 3017 году человечество наконец освоило межпланетные перелёты на таком уровне, что одним из самых популярных видов спорта стали гонки между астероидами. Через два дня состоится один из этапов Открытого кубка по межастероидным гонкам, который проводит известный популяризатор гонок Спарк.

Спарк любит считать статистику по заездам лучших участников и с удовольствием рассказывает интересные истории про разные этапы чемпионата и участников. В одном из рассказов он проговорился, что этап через два дня будет называться «Гран-При Альфа Центавры». Альфа Центавра — система хорошо изученная, и любому, кто понимает правила гонок, понятно, что участникам будет предложено долететь от астероида ICM-2017 до астероида YOY-2018.

Гонки всегда проходят в одной плоскости. Будем считать, что во время гонки астероиды не изменяют своё взаимное расположение. В плоскости, в которой проходит гонка, оба астероида можно представить как два непересекающихся и не касающихся выпуклых многоугольника, заданных координатами их вершин в порядке обхода против часовой стрелки.

Гоночный болид может стартовать и приземляться лишь перпендикулярно поверхности астероида. Для старта болида нужно выбрать точку строго внутри одной из сторон многоугольника. Стартовав из этой точки, болид летит в направлении, перпендикулярном этой стороне, удаляясь от астероида. Подлетев к другому астероиду, болид должен осуществить посадку в точку строго внутри одной из сторон многоугольника, перпендикулярно этой стороне.

Начинающий гонщик Эльбрус планирует принять участие в гонке, но он мало практиковался и не умеет маневрировать в космосе. К счастью, может оказаться, что между астероидами есть прямой маршрут и болид может долететь от одного астероида до другого, ни в какой момент времени не поворачивая. Иначе говоря, можно выбрать точки старта и финиша так, что отрезок между ними перпендикулярен сторонам, на которых лежат точки, и не проходит через внутреннюю часть астероидов. Помогите Эльбрусу выяснить, сможет ли он принять участие в гонке.

Формат входных данных

В первой строке дано целое число n — количество вершин в многоугольнике, задающем астероид ICM-2017 ($3 \leqslant n \leqslant 200\,000$). В следующих n строках даны пары чисел x_i, y_i — координаты вершин многоугольника ICM-2017 ($-10^9 \leqslant x_i, y_i \leqslant 10^9$).

В следующей строке дано целое число m — количество вершин в многоугольнике, задающем астероид YOY-2018 ($3 \leqslant m \leqslant 200\,000$). В следующих m строках даны пары чисел x'_i, y'_i — координаты вершин многоугольника YOY-2018 ($-10^9 \leqslant x'_i, y'_i \leqslant 10^9$).

Гарантируется, что оба многоугольника выпуклые, их вершины даны в порядке обхода против часовой стрелки, никакие три вершины одного многоугольника не лежат на одной прямой.

Формат выходных данных

Если можно найти прямой маршрут между многоугольниками, выведите в первой строке «`YES`», а во второй строке номера сторон астероидов ICM-2017 и YOY-2018, между которыми есть прямой маршрут.

Считайте, что сторона i соединяет вершины i и $i+1$ многоугольника, сторона n многоугольника ICM-2017 соединяет вершины n и 1, а сторона m многоугольника YOY-2018 — вершины m и 1.

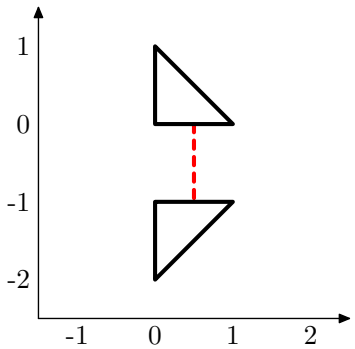
Если такого маршрута между многоугольниками нет, выведите единственное слово «`NO`».

Примеры

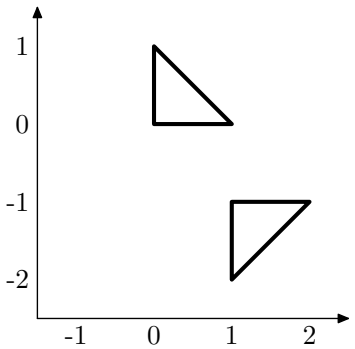
стандартный ввод	стандартный вывод
3 0 0 1 0 0 1 3 0 -1 0 -2 1 -1	YES 1 3
3 0 0 1 0 0 1 3 1 -1 1 -2 2 -1	NO
4 -1 0 0 0 0 1 -1 1 4 1 0 0 -1 1 -2 2 -1	NO

Пояснение

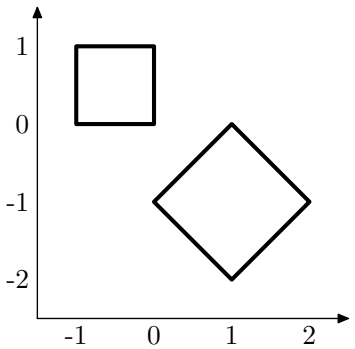
На рисунке приведены многоугольники, задающие астероиды в приведённых примерах.



Пример 1



Пример 2



Пример 3

Задача C. Конфигурационный файл

Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

Вадим разрабатывает парсер конфигурационных файлов для своего проекта. Файл состоит из блоков, которые выделяются с помощью символов «{» — начало блока, и «}» — конец блока. Блоки могут вкладываться друг в друга. В один блок может быть вложено несколько других блоков.

В конфигурационном файле встречаются переменные. Каждая переменная имеет имя, которое состоит из не более чем десяти строчных букв латинского алфавита. Переменным можно присваивать числовые значения. Изначально все переменные имеют значение 0.

Присваивание нового значения записывается как `<variable>=<number>`, где `<variable>` — имя переменной, а `<number>` — целое число, по модулю не превосходящее 10^9. Парсер читает конфигурационный файл построчно. Как только он встречает выражение присваивания, он присваивает новое значение переменной. Это значение сохраняется до конца текущего блока, а затем восстанавливается старое значение переменной. Если в блок вложены другие блоки, то внутри тех из них, которые идут после присваивания, значение переменной также будет новым.

Кроме того, в конфигурационном файле можно присваивать переменной значение другой переменной. Это действие записывается как `<variable1>=<variable2>`. Прочитав такую строку, парсер присваивает текущее значение переменной `variable2` переменной `variable1`. Как и в случае присваивания константного значения, новое значение сохраняется только до конца текущего блока. После окончания блока переменной возвращается значение, которое было перед началом блока.

Для отладки Вадим хочет напечатать присваиваемое значение для каждой строки вида `<variable1>=<variable2>`. Помогите ему отладить парсер.

Формат входных данных

Входные данные содержат хотя бы одну и не более 10^5 строк. Каждая строка имеет один из четырех типов:

- `{` — начало блока;
- `}` — конец блока;
- `<variable>=<number>` — присваивание переменной значения, заданного числом;
- `<variable1>=<variable2>` — присваивание одной переменной значения другой переменной. Переменные `<variable1>` и `<variable2>` могут совпадать.

Гарантируется, что ввод является корректным и соответствует описанию из условия. Ввод не содержит пробелов.

Формат выходных данных

Для каждой строки типа `<variable1>=<variable2>` выведите значение, которое было присвоено.

Пример

стандартный ввод	стандартный вывод

a=b
b=123
var=b
b=-34
{
c=b
b=1000000000
d=b
{
a=b
e=var
}
}
b=b
	0
	123
	-34
	1000000000
	1000000000
	123
	-34

Задача D. Совпадающие максимумы

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Саша готовится к олимпиадам по информатике и программированию и изучает структуры данных и задачи, с ними связанные. Одна из классических задач, которая попалась Саше, — поиск максимума на отрезке.

Эта задача заключается в следующем. Дан массив a из n целых чисел: $a_1, a_2, \ldots, a_n$. Требуется отвечать на запросы «найти максимум на отрезке элементов с i-го по j-й», то есть вычислить $\max\{a_i, a_{i+1}, \ldots, a_j\}$.

Конечно же, подобная задача не составила проблемы для Саши — вскоре была готова программа, которая отвечает на подобные запросы, да ещё и очень быстро. И тут выяснилось, что на разных отрезках результаты часто совпадают.

Теперь Сашу интересует, сколько же всего способов взять пару отрезков, чтобы эти отрезки не имели общих элементов, а максимумы значений элементов массива на этих двух отрезках совпадали.

Иначе говоря, требуется найти количество четвёрок i, j, k, l, таких, что $1 \leqslant i \leqslant j < k \leqslant l \leqslant n$ и $\max\{a_i, a_{i+1}, \ldots, a_j\} = \max\{a_k, a_{k+1}, \ldots, a_l\}$. Поскольку это количество может оказаться довольно большим, следует вывести остаток от деления этого числа на $1\,000\,000\,007$.

Формат входных данных

В первой строке входных данных содержится целое число n — размер Сашиного массива ($2 \leqslant n \leqslant 100\,000$).

Во второй строке содержатся n целых чисел: элементы массива. Числа в массиве положительные и не превосходят 10^9.

Формат выходных данных

Выведите остаток от деления количества пар непересекающихся отрезков, на которых совпадают максимумы, на $10^9 + 7$.

Примеры

стандартный ввод	стандартный вывод
6 3 3 4 4 3 2	16
12 1 3 2 3 4 1 3 4 3 2 2 5	177

Пояснение

Подходящие пары отрезков в первом тесте:

$[\mathbf{3}], [\mathbf{3}], 4, 4, 3, 2$ $\quad i=1 \quad j=1 \quad k=2 \quad l=2$
$[\mathbf{3}], 3, 4, 4, [\mathbf{3}], 2$ $\quad i=1 \quad j=1 \quad k=5 \quad l=5$
$[\mathbf{3}], 3, 4, 4, [\mathbf{3}, 2]$ $\quad i=1 \quad j=1 \quad k=5 \quad l=6$
$[\mathbf{3}, \mathbf{3}], 4, 4, [\mathbf{3}], 2$ $\quad i=1 \quad j=2 \quad k=5 \quad l=5$
$[\mathbf{3}, \mathbf{3}], 4, 4, [\mathbf{3}, 2]$ $\quad i=1 \quad j=2 \quad k=5 \quad l=6$
$3, [\mathbf{3}], 4, 4, [\mathbf{3}], 2$ $\quad i=2 \quad j=2 \quad k=5 \quad l=5$
$3, [\mathbf{3}], 4, 4, [\mathbf{3}, 2]$ $\quad i=2 \quad j=2 \quad k=5 \quad l=6$
$[3, 3, \mathbf{4}], [\mathbf{4}], 3, 2$ $\quad i=1 \quad j=3 \quad k=4 \quad l=4$
$[3, 3, \mathbf{4}], [\mathbf{4}, 3], 2$ $\quad i=1 \quad j=3 \quad k=4 \quad l=5$
$[3, 3, \mathbf{4}], [\mathbf{4}, 3, 2]$ $\quad i=1 \quad j=3 \quad k=4 \quad l=6$
$3, [3, \mathbf{4}], [\mathbf{4}], 3, 2$ $\quad i=2 \quad j=3 \quad k=4 \quad l=4$
$3, [3, \mathbf{4}], [\mathbf{4}, 3], 2$ $\quad i=2 \quad j=3 \quad k=4 \quad l=5$
$3, [3, \mathbf{4}], [\mathbf{4}, 3, 2]$ $\quad i=2 \quad j=3 \quad k=4 \quad l=6$
$3, 3, [\mathbf{4}], [\mathbf{4}], 3, 2$ $\quad i=3 \quad j=3 \quad k=4 \quad l=4$
$3, 3, [\mathbf{4}], [\mathbf{4}, 3], 2$ $\quad i=3 \quad j=3 \quad k=4 \quad l=5$
$3, 3, [\mathbf{4}], [\mathbf{4}, 3, 2]$ $\quad i=3 \quad j=3 \quad k=4 \quad l=6$

Задача E. Городская олимпиада

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Завтра в столице Лайнландии пройдет городская олимпиада по программированию среди школьников.

Дома в столице расположены в целочисленных точках вдоль главной улицы. Для олимпиады будет организовано три места проведения, также расположенных на главной улице. Во всех трёх местах есть ограничение на максимальное количество участников, которые могут написать олимпиаду в этом месте.

Первое место проведения находится в точке с координатой a и вмещает максимум n_a участников, второе находится в точке с координатой b и вмещает n_b участников, а третье находится в точке с координатой c и вмещает n_c участников.

В олимпиаде примет участие n школьников, i-й школьник живет в доме, который находится в точке с координатой x_i. Для каждого школьника требуется выбрать место проведения, при этом запрещается превышать максимальное количество участников в каждом месте проведения. Гарантируется, что суммарная вместительность мест проведения достаточна, чтобы разместить всех школьников.

Если школьник живет в точке с координатой p, а место проведения олимпиады, куда он должен попасть, находится в точке с координатой q, то ему придется пройти расстояние $|p - q|$ перед олимпиадой. Помогите организаторам найти минимальное суммарное расстояние, которое придётся пройти всем школьникам при их оптимальном распределении по местам проведения.

Формат входных данных

В первой строке находятся два целых числа a и n_a — координата первого места проведения и его вместительность, во второй строке находятся два целых числа b и n_b — координата второго места проведения и его вместительность, в третьей строке находятся два целых числа c и n_c — координата третьего места проведения и его вместительность ($-10^9 \leqslant a, b, c \leqslant 10^9$; $1 \leqslant n_a, n_b, n_c \leqslant 100\,000$).

В четвертой строке находится целое число n — количество школьников, которые примут участие в олимпиаде ($1 \leqslant n \leqslant 100\,000$, $n \leqslant n_a + n_b + n_c$).

В следующей строке находятся n целых чисел x_i — координаты домов, где живут участники олимпиады ($-10^9 \leqslant x_i \leqslant 10^9$).

Формат выходных данных

Выведите одно целое число — минимальное суммарное расстояние, которое придётся пройти школьникам до мест проведения.

Пример

стандартный ввод	стандартный вывод
0 1 3 2 6 3 4 -2 1 3 2	8

Задача F. Пандемия 2

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Друзья Васи постоянно играют в настольную игру «Пандемия». Васе это немного надоело, поэтому он решил создать свою версию этой игры.

На карте Байтландии он выбрал n городов, пронумерованых числами от 1 до n, и $n-1$ двухстронних дорог, соединяющих эти города. Дороги пронумерованы числами от 1 до $n-1$, i-я дорога имеет длину l_i километров и соединяет города с номерами u_i и v_i. При этом между любой парой выбранных городов существует путь вдоль выбранных дорог. В процессе игры города и дороги оказываются заражены опасной инфекцией. Город может быть полностью заражён или не заражён, а у дороги могут быть заражены некоторые участки.

По задумке Васи в начале игры происходит заражение городов с номерами $a_1, a_2, \ldots, a_m$. Затем инфекция распространяется вдоль прилежащих дорог. Как только она достигает ранее незаражённого города, он становится заражённым. В тот же момент инфекция начинает распространяться по прилежащим к только что заражённому городу дорогам. Заражение города происходит мгновенно, а по дорогам инфекция распространяется с одной и той же скоростью: один километр в минуту.

В каждый момент времени незаражённые города и участки дорог образуют *компоненты незаражённой связности*. Незаражённый город и незаражённые участки прилежащих к нему дорог всегда находятся в одной компоненте. Два незаражённых города находятся в одной компоненте в том и только в том случае, если между ними существует путь, проходящий по полностью незаражённым дорогам и городам. Компонента незаражённой связности может и вовсе не содержать городов, а только незаражённый участок дороги, которая соединяет уже заражённые города.

Игра заканчивается в тот момент, когда все города и дороги становятся полностью заражёнными. Вася еще не придумал роль игроков в этой игре, но для начала он хочет узнать, какое максимальное количество компонент незаражённой связности будет одновременно существовать в процессе игры.

Формат входных данных

Первая строка входных данных содержит целое число n — количество выбранных городов ($2 \leqslant n \leqslant 10^5$). Следующие $n-1$ строк содержат описание выбранных дорог, i-я строка содержит три целых числа u_i, v_i и l_i — номера городов, соединённых i-й дорогой, и длину этой дороги ($1 \leqslant u_i, v_i \leqslant n$; $u_i \neq v_i$; $1 \leqslant l_i \leqslant 10^9$).

Следующая строка содержит целое число m — количество городов, в которых происходит заражение в начале игры ($1 \leqslant m \leqslant n$). В следущей строке находятся целые числа $a_1, a_2, \ldots, a_m$ — номера этих городов ($1 \leqslant a_i \leqslant n$, все a_i различны).

Формат выходных данных

Выведите одно целое число — максимальное количество компонент незаражённой связности, одновременно существующих в процессе игры.

Пример

стандартный ввод	стандартный вывод
8 1 2 1 1 3 1 2 4 1 2 5 1 3 6 1 3 7 1 1 8 4 2 1 8	5

Задача G. Пазл

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Андрей планирует открыть производство деталей для пазлов. Для этого ему нужно заказать специальное устройство, которое будет вырезать детали из картона, а также приобрести набор насадок для него. Каждая насадка позволяет вырезать из заготовки деталь одной конкретной формы.

Деталь пазла представляет собой квадрат, каждая из четырех сторон которого может содержать либо выступ, либо выемку, либо быть ровной. Детали бывают трех различных видов:

- *Угловые* — у угловых деталей ровно две стороны ровные, причём это две соседние стороны, которые тем самым образуют угол.
- *Крайние* — у крайних деталей ровно одна сторона ровная.
- *Обычные* — у обычных деталей нет ровных сторон.

Выемки и выступы в деталях бывают k различных типов. Таким образом, есть $2k+1$ вариант для конкретной стороны детали: выемка одного из k различных типов, выступ одного из k различных типов, ровная сторона.

Андрею требуется понять, сколько различных насадок ему необходимо приобрести. При этом детали, одну из которых можно повернуть таким образом, чтобы она совпала с другой, можно, очевидно, вырезать с помощью одной и той же насадки.

Помогите Андрею выяснить количество различных насадок, которые необходимо приобрести, чтобы можно было вырезать деталь любой возможной формы.

Формат входных данных

Входные данные содержат целое число k — количество различных типов выемок и типов выступов у одной стороны детали ($1 \leqslant k \leqslant 10^4$).

Формат выходных данных

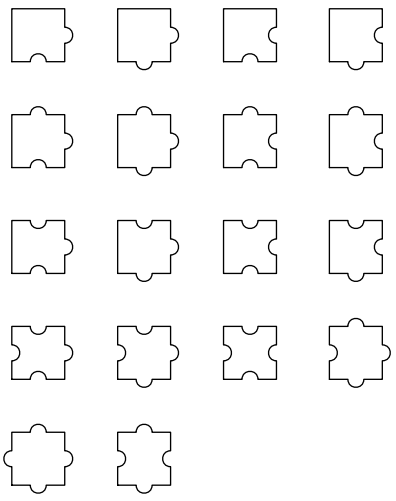
Выведите количество различных насадок, которые Андрею надо заказать.

Пример

стандартный ввод	стандартный вывод
1	18

Пояснение

Все 18 насадок для $k = 1$ представлены на рисунке.



Задача H. Игра в 2-SAT

Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Это интерактивная задача.

На экзамене по теории сложности Пете попалась следующая задача. Дано логическое выражение в виде *2-КНФ*. Задача состоит в том, чтобы определить, можно ли присвоить переменным значения `true` или `false` таким образом, чтобы значение данного выражения стало `true`.

Логическое выражение содержит переменные, каждая из которых может принимать значения `true` или `false`. Выражение в виде *2-КНФ* устроено следующим образом:

- Выражение в *2-КНФ* — это *конъюнкция* (то есть операция «и») одного или нескольких *дизъюнктов*.
- Каждый *дизъюнкт* — это операция «или», применённая к двум различным переменным или их отрицаниям.
- Переменная может входить больше чем в один дизъюнкт.
- Могут существовать переменные, которые не входят ни в один дизъюнкт.

Вот несколько примеров выражений в *2-КНФ* (символы «&», «|», «!» означают соответственно «и», «или», «не»):

- $(a|b)\&(c|!d)$
- $(!a|!b)$
- $(a|b)\&(a|!b)\&(b|!c)$

Петя решил, что ответ на задачу — «невозможно». Теперь ему предстоит доказать это учителю. Для этого учитель предложил ему сыграть в интересную игру.

В начале игры ни одной из переменных из данного выражения не присвоено значение. Петя и учитель ходят по очереди, начиная с Пети. На каждом ходу игрок выбирает любую переменную, которой ещё не присвоено значение, и присваивает ей `true` или `false`. Когда всем переменным уже присвоено значение, игра заканчивается.

Если при подстановке полученных значений переменных в данное выражение получается `false`, то считается, что Петя выиграл игру, успешно доказал свой ответ и сдал экзамен. Если же получается `true`, то ответ Пети, очевидно, ошибочен, и экзамен он не сдал.

Помогите Пете сдать экзамен. Напишите программу, которая будет играть в эту игру за Петю. Если у Пети не существует выигрышной стратегии, нужно сразу же об этом сообщить, чтобы не тратить время на проигрышную игру.

Протокол взаимодействия

Сначала на вход вашей программе подаётся выражение в виде *2-КНФ*. В первой строке заданы числа n и m — количество переменных и дизъюнктов ($2 \leqslant n \leqslant 10^4$, $1 \leqslant m \leqslant 2 \cdot 10^4$).

Далее в m строках заданы дизъюнкты. Дизъюнкт задаётся двумя числами — номерами переменых. Если переменная входит в дизъюнкт с отрицанием, то её номер задан со знаком минус. Переменные нумеруются с единицы.

Если у Пети нет выигрышной стратегии, программа должна вывести «`-1 -1`» и завершиться.

В противном случае начинается игра. Ваша программа играет за Петю, а программа жюри — за учителя. Каждый ход игрок, который его выполняет, печатает два целых числа. Присваивание переменной с номером x значения v задается строкой «`x v`». Чтобы присвоить значение `false`, необходимо вывести значение $v = 0$, а чтобы присвоить значение `true`, необходимо вывести значение $v = 1$.

Ход Пети ваша программа должна вывести в стандартный поток вывода. После каждого своего хода, если игра не закончилась, ваша программа получит в стандартный поток ввода очередной ход учителя в таком же формате.

По окончании игры ваша программа должна завершиться.

Пояснение

После каждого действия вашей программы выводите символ перевода строки. Если вы используете «`writeln`» в Паскале, «`cout << ... << endl`» в C++, «`System.out.println`» в Java или «`print`» в Python, сброс потока вывода у вас происходит автоматически, дополнительно делать «`flush`» не обязательно. Если вы используете другой способ вывода, рекомендуется делать «`flush`», но всё равно обязательно требуется выводить символ перевода строки.

Ниже приведены наиболее типичные причины получения тех или иных сообщений об ошибке.

Если ваша программа соблюдает протокол, но в конце игры значение данного выражения равно `true`, то вы получите результат «Wrong Answer».

Если ваша программа вывела вместо первого хода «`-1 -1`», но у вас была выигрышная стратегия, то вы получите результат «Wrong Answer».

Если ваша программа выводит некорректно отформатированные сообщения программе жюри, то вы получите результат «Wrong Answer».

Если ваша программа нарушила протокол и ждёт ввода в то же время, когда его ждёт и программа жюри, то вы получите результат «Idleness Limit Exceeded». Обратите внимание, что к такому же результату может привести и то, что вы не переводите строку после каждого выведенного сообщения или выводите не тем способом, который описан в начале раздела, и не делаете «`flush`».

Примеры

стандартный ввод	стандартный вывод
3 2 1 2 -1 -2 2 0	 3 1 1 0
2 2 1 2 -1 -2	 -1 -1

Задача I. Электрическая цепь

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

На практических занятиях по физике в школе Женя проходит электричество. Очередная лабораторная работа состоит в анализе электрической цепи.

Электрическая цепь представляет собой набор узлов, соединенных проводами. Один из узлов является входом, а некоторый другой узел является выходом. При этом электрическая цепь в задании является *хорошей*.

Хорошей называется цепь, построенная по следующим правилам:

- Базовая цепь, состоящая только из входа и выхода, соединённых одним проводом, является хорошей.
- Если есть две хороших цепи A и B, то цепь $S(A, B)$, составленная из них объединением выхода цепи A со входом цепи B, является хорошей. Входом получившейся цепи является вход цепи A, а выходом — выход цепи B.
- Если есть две хороших цепи A и B, то цепь $P(A, B)$, составленная из них объединением входа A со входом B, а также выхода A с выходом B, является хорошей. Входом получившейся цепи является объединённый вход, а выходом — объединённый выход.
- Любая хорошая цепь может быть построена из базовых применением конечного числа предыдущих двух правил.

На рисунке приведены примеры хороших цепей.

Базовая цепь Цепь из примера 1 Цепь из примера 2 Цепь из примера 3

Другие примеры хороших цепей

Женя не очень любит физику, но очень любит теорию графов. Поэтому вместо выполнения задания, он хочет посчитать количество способов убрать из цепи некоторые провода так, чтобы оставшийся набор проводов образовывал *дерево*: от каждого узла до каждого существовал путь по проводам, причем единственный.

Так как количество может быть слишком большим, Женя хочет найти остаток от его деления на 998 244 353. Помогите Жене.

Формат входных данных

В первой строке дано два целых числа n и m — число узлов в сети и число проводов, соответственно ($1 \leqslant n, m \leqslant 100\,000$).

В следующих m строках дано описание проводов. Каждая из них содержит два целых числа a и b, которые обозначают, что узлы a и b соединены проводом ($1 \leqslant a, b \leqslant n$; $a \neq b$).

Гарантируется, что заданная цепь является хорошей. Входом цепи является узел с номером 1, а выходом — узел с номером n. Обратите внимание, что в хорошей цепи несколько проводов могут соединять одну и ту же пару узлов.

Формат выходных данных

Выведите одно целое число — остаток от деления искомого количества способов на 998 244 353.

Примеры

стандартный ввод	стандартный вывод
3 3 1 2 2 3 3 1	3
6 7 1 2 1 3 1 6 2 4 3 5 4 6 5 6	15
2 2 1 2 1 2	2

Пояснение

В первом и третьем примерах можно убрать один любой провод.

Во втором примере можно либо убрать провод $(1, 6)$ и любой другой, либо по одному проводу из цепочек $1 - 2 - 4 - 6$ и $1 - 3 - 5 - 6$.

Задача J. Развлечение с копьями

Ограничение по времени: 3 секунды
Ограничение по памяти: 512 мегабайт

В Арифметическом университете иногда бывает так, что занятия по арифметике отменяются. Чтобы студенты не скучали без дела, для них организовали развлечение — метание копья. Для этого в комнате отдыха разместили автомат для продажи копий и мишень для метания.

Мишень состоит из нескольких слоёв, и каждое копьё при броске пробивает какое-то количество этих слоёв. Конечная цель — пробить мишень насквозь.

Каждое копьё характеризуется своими диаметром, прочностью и ценой. Автомат предлагает покупать копья по одному в определённой последовательности. Каждое предложенное копьё можно либо купить и сразу бросить в мишень, либо отказаться от его покупки, тогда автомат предлагает купить следующее копьё. После того, как автомат предожил для покупки все имеющиеся копья, он отключается.

Первое брошенное копьё оставляет в мишени круглую дыру с диаметром, равным диаметру брошенного копья, и глубиной, равной прочности брошенного копья. Каждое следующее копьё бросается точно в центр образованной дыры. Если диаметр этого копья не превосходит диаметров всех предыдущих брошенных в мишень копий, то оно продолжает пробивать слои мишени, начиная с первого слоя, в котором пока нет дыры. Иначе оно начинает пробивать слои, начиная с того слоя, диаметр дыры в котором строго меньше диаметра копья.

Копьё пробивает количество слоёв равное своей прочности. Количество пробитых слоёв не зависит от того, были ли они уже пробиты ранее копьём меньшего диаметра. Если копьё имеет достаточную прочность, чтобы пробить последний слой мишени, оно пролетает мишень насквозь и игра завершается.

Вам известна последовательность, в которой автомат предлагает купить копья. Определите какие копья нужно купить и бросить, чтобы пробить все слои мишени, заплатив при этом минимальную возможную сумму.

Формат входных данных

В первой строке даны два целых числа n и m — количество копий в автомате и количество слоёв мишени ($1 \leqslant n, m \leqslant 2000$).

В следующих n строках описаны копья. В i-й из этих строк даны три числа d_i, c_i и p_i — диаметр, прочность и цена i-го копья в последовательности, предлагаемой автоматом ($1 \leqslant d_i, p_i \leqslant 10^9$; $1 \leqslant c_i \leqslant 2000$).

Формат выходных данных

В первой строке выведите два целых числа s и k — суммарную цену копий в оптимальном наборе и их количество. В следующей строке выведите k целых чисел — номера взятых копий. Номера следует выводить в возрастающем порядке, именно в этом порядке копья будут куплены и брошены в мишень.

Если не существует подходящего набора, выведите одно число «-1». Если существует несколько наборов копий с минимальной суммарной ценой, выведите любой из них.

Примеры

стандартный ввод	стандартный вывод
2 2 1 1 1 2 3 2	2 1 2
2 4 1 1 1 2 3 2	-1
2 4 1 1 1 1 3 2	3 2 1 2

Задача K. Смишенное произведение

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Миша очень любит изобретать новые операции. Недавно он изобрёл новую операцию, которую он в честь себя назвал *смишенным произведением*.

Смишенное произведение набора различных натуральных чисел $a_1, a_2, \ldots, a_n$ устроено следующим образом. Рассмотрим все возможные упорядоченные пары (a_i, a_j) различных чисел этого набора. Для каждой пары запишем эти числа подряд без пробела, получив новое число b_{ij}. Смишенным произведением чисел из исходного набора Миша называет сумму всех значений b_{ij}.

Помогите Мише посчитать смишенное произведение заданного набора чисел. Миша хочет вычислить его по модулю $10^9 + 7$.

Формат входных данных

Первая строка содержит одно натуральное число n — количество чисел в наборе ($2 \leqslant n \leqslant 10^5$).

Вторая строка содержит n различных натуральных чисел $a_1, a_2, \ldots, a_n$ — числа набора ($1 \leqslant a_i \leqslant 10^8$).

Формат выходных данных

Выведите одно число — остаток от деления смишенного произведения заданных чисел на число $10^9 + 7$.

Пример

стандартный ввод	стандартный вывод
3 1 3 10	668

Пояснение

В примере из условия есть шесть возможных пар, получаются следующие значения b_{ij}: 13, 31, 101, 103, 110, 310, их сумма равна 668.

Задача L. Простые суффиксы

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Будем называть *суффиксом* числа x число y, которое получается из десятичной записи x откидыванием любого числа первых цифр. Если в десятичной записи x не встречается нулей, то все суффиксы x не содержат ведущих нулей. Например, суффиксами числа 283 являются числа 283, 83 и 3.

Число называется простым, если оно имеет ровно два натуральных делителя. Заметим, что число 1 простым не является — у него только один натуральный делитель.

Сене нравятся простые числа, не содержащие нулей в десятичной записи, все суффиксы которых также являются простыми числами.

Заданы целые числа a и b. Помогите Сене подсчитать, сколько целых чисел между a и b включительно ему нравится.

Формат входных данных

Входные данные содержат два целых числа a и b ($1 \leqslant a \leqslant b \leqslant 10^{11}$).

Формат выходных данных

Выведите количество простых чисел, не содержащих нулей, от a до b включительно, таких, что если откинуть сколько угодно первых цифр числа, то оставшееся число всё ещё будет простым.

Примеры

стандартный ввод	стандартный вывод
4 13	3
101 109	0
281 286	1

Пояснение

В первом примере подходят числа 5, 7 и 13.

Во втором примере ни одно число не подходит, так как все числа в диапазоне содержат 0.

В третьем примере число 283 подходит, так как числа 283, 83 и 3 — простые.

Задача M. Восстановление последовательности

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

В первом классе сегодня проходят натуральные числа. Учитель выписал на доске в возрастающем порядке числа от 1 до n, чтобы показать их школьникам.

К сожалению, хулиган Коля из 11 класса вбежал в класс и испортил содержимое доски. Оказавшись в кабинете директора, он раскаялся и сказал, что всего лишь стёр с доски одно из чисел.

Помогите учителю разобраться, не врёт ли Коля, и если он действительно всего лишь стёр одно число, то какое это было число.

Формат входных данных

Первая строка входных данных содержит число n — количество чисел, которое учитель написал на доске ($2 \leqslant n \leqslant 1000$).

Вторая строка входных данных содержит число m — количество чисел на доске после Колиных хулиганских действий ($1 \leqslant m \leqslant 1000$).

Третья строка содержит m целых чисел $a_1, a_2, \ldots, a_m$ — числа, которые оказались на доске после Колиных действий, в том порядке, в котором они записаны на доске ($1 \leqslant a_i \leqslant 1000$).

Формат выходных данных

Если Колины объяснения правдоподобны, и числа, выписанные на доске после Колиных действий, могли получиться из последовательности $1, 2, \ldots, n$ стиранием одного числа, выведите в первой строке слово «Yes». Во второй строке выведите число, которое стёр Коля.

В противном случае выведите в первой строке слово «No».

Примеры

стандартный ввод	стандартный вывод
4 3 1 3 4	Yes 2
4 3 3 3 3	No
4 2 1 2	No
4 4 1 2 3 4	No
4 3 4 3 1	No