Problem A. Количество единиц

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

Вася учится в 11 классе и собирается сдавать ЕГЭ по информатике. К сожалению, ему с трудом даются задания, в которых надо посчитать количество единиц в двоичном представлении значения выражения $2^a + 2^b - 2^c$. Вася выписал несколько подобных заданий и решил их. Теперь он хочет проверить свои решения и просит вас написать для этого программу, которая по заданным a, b и c найдет ответ.

Input

В первой и единственной строке ввода через пробел перечислены три целых числа a, b и c ($1 \le c < b < a \le 10^9$).

Output

Вывести количество единиц в двоичной записи значения выражения $2^a + 2^b - 2^c$.

Example

стандартный ввод	стандартный вывод
3 2 1	2

Problem B. Кратеры

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

Поверхность Луны — усеянная множеством кратеров пустыня. За миллионы лет её случайным образом бомбардировали тысячи метеоритов.

Сейчас к Луне подлетает аппарат, на борту которого находятся три датчика, которые будут сброшены на определённый участок поверхности. Для простоты будем считать этот участок квадратом на плоскости, а кратеры — точками в нём. Из-за возможных помех от космического, солнечного и других излучений, их нужно поместить в кратеры. Спутник будет пролетать мимо Луны и в отведённый временной интервал он должен сбросить датчики, чтобы они упали в нужные кратеры.

К сожалению, в самый последний момент обнаружилась неисправность: из памяти аппарата стёрлись координаты выбранных кратеров. Вам поручено вычислить их заново.

Датчики будут передавать данные об участке лунной поверхности, попадающем в треугольник с вершинами в кратерах, в которых находятся датчики, поэтому необходимо разместить датчики так, чтобы площадь участка была как можно больше.

Вычислите, в каких кратерах нужно расположить датчики.

Input

Первая строка содержит целое число N — количество кратеров ($3 \le N \le 2 \times 10^5$).

В следующих N строках идёт описание координат кратеров — пара целых чисел x_i и y_i.

Гарантируется, что каждая составляющая координат кратеров является случайным числом, выбранным равновероятно из отрезка $[-N, N]$.

Output

Выведите три строки. В каждой строке должны содержаться координаты кратера, в который будет помещён датчик. Если решений несколько, выведите любое.

Example

стандартный ввод	стандартный вывод
5 1 5 -2 5 -1 2 4 -3 5 3	4 -3 5 3 -2 5

Problem C. Победитель

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

В Берляндии проходит олимпиада по программированию. Оргкомитет соревнований хочет провести красивую церемонию закрытия и вывести название команды-победителя соревнований на большой экран. К сожалению, после проведения олимпиады все организаторы будут заняты более важными делами и название команды будет определять некому.

Поэтому было решено написать программу, которая в нужный момент считает информацию о соревнованиях из тестирующей системы, определит победителя и выведет название этой команды на экран. Организаторы попросили вас помочь в написании этой программы.

Правила определения победителя достаточно просты и вам знакомы: каждая команда по окончании олимпиады имеет результат, который выражается двумя числами – количеством решенных задач и штрафным временем. Победителем соревнований будет команда, которая решит наибольшее количество задач. А если таких команд будет несколько, то среди них выбирается команда с наименьшим штрафным временем. Гарантируется, что в данном соревновании не найдется двух команд с идентичными результатами (одинаковое количество задач и одинаковое время).

Input

Входные данные содержат данные о результатах соревнований, взятые из тестирующей системы. В первой строке записано одно натуральное число n ($n \le 100$) – количество команд. Далее идет $n \times 2$ строк. В строке с номером $i \times 2$ содержится название i-й команды (не менее одного и не более 20 символов, все символы – строчные и прописные латинские буквы. Гарантируется, что все названия различны). В строке с номером $i \times 2 + 1$ содержатся два целых числа: p_i ($0 \le p_i \le 15$) и t_i ($0 \le t_i \le 5000$) – количество решенных задач и штрафное время для i-й команды. Порядок, в котором заданы команды, не обязан совпадать с порядком, в котором команды идут в отсортированной таблице результатов.

Output

Выведите единственную строку – название команды-победителя.

Example

стандартный ввод	стандартный вывод
5 Harvard 10 1358 MIPT 10 1437 Shanghai 11 1567 SPbSU 11 1560 Warsaw 10 1586	SPbSU

Problem D. Небоскребы

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

В городе Exo есть ровно N небоскребов, которые расположены друг за другом в один ряд. Если смотреть на них слева направо, то их высоты равны: $a_1, a_2, \ldots, a_N$.

В город вторглось ужасное чудовище, имя которому: «Угурбато». Каждым своим ударом оно разрушает один из небоскребов ударом такой силы, что влево и вправо от него расходится ударная волна.

Рассмотрим левую ударную волну. Пусть чудовище ударило по небоскребу под номером i. Если на пути волны встретился уже разрушенный небоскреб, то волна затухает и дальше не идет. Если волна проходит через целый небоскреб под номером k, то он разрушается только в том случае, если $a_i - a_k \geq |i - k|$ (Если небоскреб разрушится, то волна все равно пойдет дальше). Аналогичные утверждения справедливы и для правой ударной волны.

Чудовище наносит всего M ударов, причем оно совершает новый удар только тогда, когда обе ударные волны от предыдущего удара затухли. Мэр города Exo просит вас посчитать сколько небоскребов было разрушено после каждого удара Угурбато.

Input

Первая строка содержит одно целое число N ($1 \leq N \leq 10^5$) — количество небоскребов.

Вторая строка содержит N целых чисел $a_1, a_2, ..., a_N$, разделённых пробелами — высоты небоскребов, перечисленные слева направо ($1 \leq a_i \leq 10^9$).

Третья строка содержит одно целое число M ($1 \leq M \leq N$) — количество ударов чудовища.

Четвертая строка содержит M различных чисел через пробел: $b_1, b_2, ..., b_M$ — номера небоскребов, которые разрушало чудовище в заданном порядке (небоскребы нумеруются слева направо, начиная с единицы). Гарантируется, что никакой небоскреб под номером $b_i (1 \leq i \leq M)$ не был разрушен ударной волной.

Output

Выходные данные должны содержать M строк, где в i-ой строке содержится единственное целое число — количество небоскребов, которые были разрушены после i-го удара чудовища.

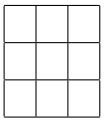
Examples

стандартный ввод	стандартный вывод
5 3 2 4 10 5 2 3 4	2 2
5 1 2 3 2 1 1 3	5

Problem E. Доска

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

Вася учится в первом классе математической школы и делает большие успехи в освоении математики. Сегодня учитель задал детям следующую задачу. Сначала он расчертил доску так, чтобы она представляла собой матрицу $N \times N$, как показано на рисунке (в данном примере $N = 3$).



Далее, учитель продемонстрировал различные варианты, как можно заполнить матрицу числами от 1 до N^2 строго по возрастанию:

(1)

1	2	3
6	5	4
7	8	9

(2)

1	6	7
2	5	8
3	4	9

(3)

1	2	3
8	9	4
7	6	5

(4)

1	8	7
2	9	6
3	4	5

Учитель всегда начинал с верхнего левого угла. В матрице (1) пример представлял собой вертикальную «змейку», в матрице (2) — горизонтальную. Матрицы (3) и (4) представляют собой спираль — в примере (3) заполнение происходит по часовой стрелке, а в примере (4) — против часовой.

Вам необходимо помочь Васе написать программу, которая по заданному N и алгоритму заполнения — (1), (2), (3) или (4) будет печатать результирующую матрицу.

Input

В единственной строке содержатся два натуральных числа N ($1 \le N \le 100$) и a ($1 \le a \le 4$), где a определяет алгоритм заполнения матрицы.

Output

Выведите N строк, состоящих из N чисел, разделенных пробелами, представляющих собой матрицу, заполненную по заданному алгоритму.

Examples

стандартный ввод	стандартный вывод
3 1	1 2 3 6 5 4 7 8 9
3 2	1 6 7 2 5 8 3 4 9
3 3	1 2 3 8 9 4 7 6 5
3 4	1 8 7 2 9 6 3 4 5

Note

В примерах даны входные и выходные данные матриц, предложенных учителем.

Problem F. Числа-друзья

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

У Поликарпа много друзей. Ему это доставляет большое удовольствие, но недавно он задумался, а есть ли друзья у обычных целых чисел?

Поликарп называет два числа *друзьями*, если одно из них делится на другое без остатка.

Например, 2 и 4 будут таковыми, а 10 и 3 нет. В частности, число 1 дружит со всеми.

Поликарпу интересно, можно ли выстроить все целые числа от 1 до n в ряд так, чтобы соседние числа были друзьями? Найдите такую расстановку или сообщите, что её не существует.

Input

В единственной строке содержится целое число n ($1 \le n \le 1000$).

Output

Если ответа не существует, выведите одно число «-1» (без кавычек).

Иначе, выведите n чисел в одной строке через пробел — найденый ряд из n различных чисел.

Если ответов несколько, разрешается вывести любой.

Examples

стандартный ввод	стандартный вывод
2	2 1
3	3 1 2

Problem G. Гмугл

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

В рамках программы импортозамещения Вам поручена разработка нового поисковика под кодовым названием Гмугл (gmoogle). Для первого прототипа Вам требуется показать возможность поиска по базе данных предложений:

- Тексты базы данных соединены в строку S, состоящую из символов 'a'-'z', 'A'-'Z', пробелов, знаков препинания «.!?» (без учета кавычек) а также цифр.
- Символы «.!?» разрывают S на одно или более предложений с одним исключением: если первый символ после точки '.', не являющийся пробелом — буква в нижнем регистре ('a'-'z'), то такая точка предложение не разрывает (пример: «I like tea in a 500 ml. cup» — это одно предложение, а «Cup is 500 ml. I want it» и «Cup is 500 ml. 500 ml is great for me» содержат по два предложения).
- *Словом* называется непрерывная последовательность символов 'a'-'z', 'A'-'Z', ограниченная справа и слева пробелами, знаками препинания или началом/концом строки/предложения. Цифры вплотную к слову находиться не могут, то есть конструкции «10ml» или «R2D2» являются некорректными и в строке S не встретятся.
- В S могут встречаться предложения, не содержащие слов. Гарантируется, что символы «.!?» никогда не находятся рядом друг с другом.

После индексации контента поисковиком пользователи делают поисковые запросы в виде строки q, состоящей из одного или более слов (определение слова дано выше), разделенных пробелами. В начале или конце запроса также могут быть пробелы.

Поисковик должен найти и выдать предложения из базы данных S, где присутствуют все слова из запроса q в любом порядке. Равенством слов считается совпадением всех их букв без учета регистра.

Input

В первой строке входа содержится строка S длиной от 1 до 1000 символов. В следующей строке задано целое число n ($1 \le n \le 100$), — количество поисковых запросов. Далее следуют n строк $q_1, q_2, ..., q_n$, каждая длиной от 1 до 100 символов, в формате, описанном выше. Между, перед и после слов **может** быть любое количество пробелов — как и в строке S.

Output

Для каждого поискового запроса $q_1, q_2, ..., q_n$ поисковик выводит сам запрос на отдельной строке. Далее, каждое в своей строке, выводится список найденных предложений из S, в том порядке, в котором они присутствуют в S. Поисковые запросы и предложения печатаются в кавычках.

В начале и конце предложений пробелы **не печатаются**. См. пример для более точного понимания.

Example

стандартный ввод	стандартный вывод
Hello everyone. I want 2 coffee if you have it. I like coffee very much. 4 HELLO Coffee much coffee VoDka	Search results for "HELLO": - "Hello everyone." Search results for "Coffee": - "I want 2 coffee if you have it." - "I like coffee very much." Search results for "much coffee": - "I like coffee very much." Search results for "VoDka":

Problem H. Генератор

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	256 мегабайт

Поликарп и его коллеги празднуют завершение работы над инновационной системой защиты от кибератак. Над системой трудилось множество людей, Поликарп был ответственным за криптографическую часть. Одним из алгоритмов, разработанных им, является поиск случайного простого числа, не превосходящего N. Напомним, что простым числом является число, имеющее ровно два различных делителя — единица и само число.

Алгоритм поиска простого числа, написанный Поликарпом, выглядит так:

1. Выбрать случайное число x из отрезка $[2, N]$. Для всех $N-1$ чисел вероятности быть выбранными одинаковы.
2. Проверить, является ли x простым. Если да, перейти к шагу 3, иначе к шагу 1.
3. Возвратить x как результат.

Проверку того, является ли число x простым, Поликарп реализовал отдельно. Он пользовался тем, что для составного числа всегда существует делитель, не превосходящий квадратного корня из x и при этом отличный от единицы. Это позволяет сэкономить вычисления.

Алгоритм проверки реализован следующим образом:

1. Положить $d := 2$.
2. Если d превосходит квадратный корень из x, то есть $d^2 > x$, то завершить работу алгоритма и сообщить, что x — простое число. Иначе перейти к шагу 3.
3. Проверить, делится ли x на d без остатка. Если да, завершить проверку, сообщив, что x — не простое. Иначе положить $d := d + 1$ и перейти к шагу 2.

Но сегодня Поликарп не мог уснуть. Его беспокоила мысль о том, что его алгоритм поиска может работать вечно! Или хотябы просто долго. А ведь это ставит под угрозу работу их системы.

Самой тяжёлой операцией Поликарп считает деление в проверке на простоту на третьем шаге. Посчитайте математическое ожидание количества этих делений по заданному N.

Input

В первой строке содержится целое число T — количество тестовых примеров ($1 \le T \le 10^5$).

В каждой из следующих T строк содержится один тестовый пример — целое число N ($2 \le N \le 10^7$).

Output

На каждый тест выведите ответ в отдельной строке в виде несократимой дроби.

Example

стандартный ввод	стандартный вывод
6 2 3 4 5 6 10	0/1 0/1 1/2 2/3 1/1 2/1

Problem I. Сложение

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	5 секунд
Memory limit:	256 мегабайт

Это интерактивная задача.

Инженер Поликарп работает в компании «Абак». Каждый день он должен придумывать N двоичных строк длины M. В один из таких дней он решил разнообразить свой рабочий процесс: после того как Поликарп придумывает очередную строку, он интерпретирует её как двоичное число (возможно, с ведущими нулями), поочерёдно складывает его с каждым ранее придуманным числом, и подсчитывает сколько раз результат сложения не поместился в двоичное число длины M.

Поликарп очень плохо складывает числа, поэтому он просит вас помочь ему.

Interaction Protocol

В первой строке программа жюри передает два целых числа N и M ($1 \le N \times M \le 10^5$) — количество придуманных Поликарпом двоичных строк и длину этих строк соответственно.

Далее N раз происходит следующее: программа жюри в отдельной строке передаёт вашей программе очередное придуманное Поликарпом двоичное число, состоящее ровно из M бит, а ваша программа в отдельной строке должна выдать одно целое число — количество раз, когда при сложении данного числа и ранее переданных вам двоичных чисел результат не поместился в двоичное число длины M.

После обработки N запросов ваша программа обязана завершиться с кодом завершения 0. В противном случае результатом взаимодействия будет ошибка (даже если все ответы, выданные вашей программой, были правильными).

Examples

стандартный ввод	стандартный вывод
5 2 01 10 11 00 10	0 0 2 0 2
5 3 110 001 101 110 011	0 0 1 2 3

Note

Для корректной работы программы после каждой операции вывода данных вам необходимо очищать буфер вывода, то есть делать следующие операции:

- В языке Pascal: **flush(output)**;
- В C/C++: **fflush(stdout)** или **cout.flush()**;
- В Java: **System.out.flush()**;
- В Python: **sys.stdout.flush()** из библиотеки **sys**;

- В C#: **Console.Out.Flush**();

Также не забудьте выводить перевод строки после каждого ответа вашей программы.

Problem J. Добрый маг

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	2 секунды
Memory limit:	256 мегабайт

На поле битвы столкнулись армии добра и зла. В ходе долгого кровопролитного противостояния практически все силы добра были уничтожены. Остался лишь один маг, которому теперь предстоит в одиночку сражаться против армии из n злых существ. К счастью, добрый маг знает m заклинаний, способных уничтожать вражеских существ.

Каждое заклинание способно убить определенное множество врагов, при этом потратится определенное количество магической энергии. Есть две важных особенности использования заклинаний, которые нужно знать:

1. Заклинание можно использовать лишь в том случае, если все существа, на которые оно действует, живы до момента использования.

2. Заклинание после использования убьет всех существ, на которые действует. У мага нет возможности пощадить существо, на которое распространяется заклинание.

Сможет ли добрый маг уничтожить всех врагов и принести победу силам добра? Если да, то какое минимальное количество магической энергии для этого потребуется?

Input

В первой строке содержатся два целых числа: n ($1 \le n \le 18$) – количество злых существ и m ($0 \le m \le 100$) – количество заклинаний.

Следующие m строк содержат информацию о заклинаниях. Первое число строки содержит целое число k_i ($1 \le k_i \le n$) – количество существ, на которых действует i-е заклинание. Далее идет k_i целых чисел от 1 до n – номера существ, на которые действует заклинание (будем считать, что злые существа пронумерованы от 1 до n), числа в этом списке не повторяются. Последним в строке идет натуральное число v_i ($v_i \le 1000$) – количество магической энергии, необходимое для использования заклинания.

Output

Выведите единственное число – минимальное количество энергии, которое нужно потратить, чтобы уничтожить всех злых существ. Если всех существ убить невозможно, выведите -1.

Examples

стандартный ввод	стандартный вывод
5 6 2 1 2 10 3 3 4 5 18 2 4 5 6 1 3 7 1 2 4 1 1 11	23
3 2 2 1 2 5 2 2 3 5	-1

Note

В первом примере есть 4 способа убить всех существ: (заклинания 1 и 2 – 28 ед. энергии), (1,3,4 – 23), (2,5,6 – 33), (3,4,5,6 – 28). Лучше всего воспользоваться заклинаниями 1,3 и 4.

Во втором примере любое из заклинаний убивает существо 2. Поэтому применить можно только одно из них, но этого будет недостаточно.

Problem K. Проверка маршруток

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

Поликарп приехал в Байтогорск, один из городов Берляндии, по распоряжению начальства. Его задача — проверка маршруток города.

Дорожная сеть общественного транспорта Байтогорска состоит из N остановок и нескольких дорог с двусторонним движением, соединяющих их. Сеть примечательна тем, что между любыми двумя остановками существует единственный путь из дорог, по которому можно проехать от одной остановки к другой (в обоих направлениях). Все дороги имеют одинаковую длину; маршрутка преодолевает дорогу за одну минуту.

Так как Байтогорск очень развитый город, для каждой пары остановок существует своя компания, занимающаяся перевозкой пассажиров. Причём маршрутка совершает остановки только на конечных пунктах пути, пропуская остальные остановки, если таковые встречаются в пути. Поликарпу нужно проверить каждую компанию, проехав единожды на маршруте каждой из них, неважно в каком направлении.

Его сейчас интересует время, которое он проведёт сидя (или стоя) в маршрутках. Ему кажется, что это займёт много времени, посчитайте количество минут, которое необходимо для проезда на всех маршрутах Байтогорска, пока он морально готовится.

Input

В первой строке содержится целое число N — количество остановок в городе ($1 \le N \le 2 \times 10^5$).

В следующих строках идёт список дорог дорожной сети Байтогорска. Каждая дорога задаётся в отдельной строке парой чисел a и b ($1 \le a, b \le N; a \ne b$). Гарантируется, что каждая пара чисел указана во входе не более одного раза.

Output

Выведите в единственной строке одно число — суммарное время, необходимое для проезда на всех маршрутах города.

Examples

стандартный ввод	стандартный вывод
5 1 2 1 3 3 4 3 5	18
2 2 1	1

Problem L. Уравнение Фибоначчи

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

Бизон Миша взял три последовательных числа Фибоначчи: F_n, F_{n+1} и F_{n+2}, изменил их порядок и подставил в качестве коэффициентов квадратного уравнения:

$$Ax^2 + Bx + C = 0$$

Теперь Миша хочет узнать, сколько существует различных вещественных корней у данного уравнения, и просит вас помочь ему.

Input

В единственной строке содержится три целых неотрицательных числа: i, j и k $(i, j, k \le 10^9)$ — номера членов последовательности Фибоначчи, где $A = F_i$, $B = F_j$ и $C = F_k$. Гарантируется, что все три числа i, j, k различны и что среднее по величине отличается как от большего, так и от меньшего на единицу.

Output

В единственную строку выведите одно целое число — количество различных вещественных корней уравнения.

Examples

стандартный ввод	стандартный вывод
1 2 0	2
1 0 2	0

Note

Последовательность Фибоначчи строится по следующим правилам:

$$F_0 = 0$$

$$F_1 = 1$$

$$F_i = F_{i-1} + F_{i-2}, \text{ где } i > 1$$