Problem A. Problem Order

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

Перед тем, как задачи Moscow Programming Contest были отправлены в печать, жюри упорядочило задачи по возрастанию сложности, так что эта задача — самая простая, а задача K — самая сложная.

Алиса и Боб — участники одной из команд, прямо сейчас принимающей участие в том же самом контесте, что и вы. Их команда решила все задачи, кроме самой последней. Сейчас третий участник дописывает код последней задачи, Боб решил навести порядок на столе и сложил условия всех задач в стопку. Алиса перебирает стопку и грустно вздыхает каждый раз, когда название задачи начинается с буквы, которая идёт в алфавите раньше, чем буква, с которой начинается название предыдущей задачи.

Подсчитайте, сколько раз Алиса грустно вздохнёт во время просмотра собранной Бобом стопки.

Input

На вход подаётся список названий задач **в этом контесте** в том порядке, в котором они даны в наборе.

Output

Выведите одно число — количество раз, которое вздохнёт Алиса.

Example

стандартный ввод	стандартный вывод
Problem Order Interactor Signals in the Space PalINTdromes Ugly Polyomino Robot in the Maze DHCP Troubles Array Test Favorite Points Thorny Graph Xor and Segments	3

Note

Ответ к примеру **неверен** и приведён только для того, чтобы проиллюстрировать формат ввода-вывода.

Problem B. Interactor

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

Лена руководит разработкой тестирующей системы, в которой реализованы интерактивные задачи.

До завершения очередной стадии проекта осталось написать модуль, определяющий *итоговый вердикт* системы для интерактивной задачи. *Итоговый вердикт* определяется из кода завершения задачи, вердикта интерактора и вердикта чекера по следующим правилам:

- Вердикт чекера и вердикт интерактора — это целые числа от 0 до 7 включительно.
- Код завершения задачи — это целое число от -128 до 127 включительно.
- Если интерактор выдал вердикт 0, итоговый вердикт равен 3 в случае, если программа завершилась с ненулевым кодом, и вердикту чекера в противном случае.
- Если интерактор выдал вердикт 1, итоговый вердикт равен вердикту чекера.
- Если интерактор выдал вердикт 4, итоговый вердикт равен 3 в случае, если программа завершилась с ненулевым кодом, и 4 в противном случае.
- Если интерактор выдал вердикт 6, итоговый вердикт равен 0.
- Если интерактор выдал вердикт 7, итоговый вердикт равен 1.
- В остальных случаях итоговый вердикт равен вердикту интерактора.

Ваша задача — реализовать этот модуль.

Input

Входной файл состоит из трёх строк. В первой задано целое число r ($-128 \le r \le 127$) — код завершения задачи, во второй — целое число i ($0 \le i \le 7$) — вердикт интерактора, в третьей — целое число c ($0 \le c \le 7$) — вердикт чекера.

Output

Выведите одно целое число от 0 до 7 включительно — итоговый вердикт системы.

Examples

стандартный ввод	стандартный вывод
0 0 0	0
-1 0 1	3
42 1 6	6
44 7 4	1
1 4 0	3
-3 2 4	2

Problem C. Signals in the Space

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

Межзвёздная автоматическая станция передала на Землю закодированное тестовое сообщение, состоящее из N сигналов — целых неотрицательных чисел, не превосходящих 255 (числа в сообщении могут повторяться). Но из-за ошибки в программе с принимающей стороны числа в сообщении были переставлены, а из-за зашумлённости канала связи некоторые сигналы были распознаны некорректно.

По исходному и принятому сообщениям найдите минимально возможное количество сигналов, которые были распознаны некорректно.

Input

Первая строка входных данных содержит одно целое число N — длину сообщения ($1 \le N \le 1000$).

Во второй строке через пробел перечислены N целых неотрицательных чисел, не превосходящих 255 — отправленные станцией сигналы.

В третьей строке через пробел перечислены N целых неотрицательных чисел, не превосходящих 255 — принятые на Земле сигналы.

Output

Выведите одно целое число — наименьшее количество сигналов, которые были распознаны некорректно.

Examples

стандартный ввод	стандартный вывод
5 1 2 3 4 5 5 2 3 1 4	0
5 1 1 1 1 1 1 2 3 4 5	4

Problem D. PalINTdromes

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

Число является *существенно палиндромическим* в системе счисления с основанием $b \geq 2$, если запись этого числа в соответствующей системе без ведущих нулей состоит **более, чем из одной цифры** и является палиндромом.

Например, число 5 является существенно палиндромическим в системе счисления с основанием 2 (запись 101 является палиндромом), а числа 1 и 2 — не являются (запись 10 палиндромом не является, а запись числа 1 состоит из одной цифры); число 901684 является существенно палиндромическим в системе счисления с основанием 99 (так как $901684 = 99 \cdot 99 \cdot 91 + 99 \cdot 98 + 91$, то число состоит из цифр (91) в первом разряде, (98) во втором и (91) в третьем и тем самым запись является палиндромом).

По заданному числу n требуется найти **максимальное** целое число $b \geq 2$ такое, что в системе счисления с основанием b число n является существенно палиндромическим.

Input

Входные данные содержат одно целое число n ($3 \leq n \leq 10^9$). Гарантируется, что входные данные подобраны таким образом, что ответ всегда существует.

Output

Выведите максимальное основание b системы счисления, в которой число n является существенно палиндромическим.

Example

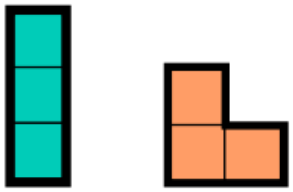
стандартный ввод	стандартный вывод
3	2

Problem E. Ugly Polyomino

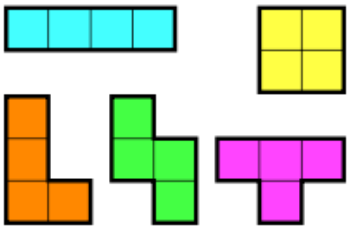
Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	1 секунда
Memory limit:	256 мебибайт

Полимино — это связная фигура из n равных квадратов, соединённых сторонами. Неформально, это связное множество клеток, которое можно вырезать из бесконечного листа клетчатой бумаги. Два полимино считаются совпадающими, если одно из них можно перевести в другое с помощью параллельных переносов, поворотов или отражений относительно прямой, параллельной одной из осей координат (которым параллельны все стороны всех квадратиков). Набор n-полимино считается полным, если никакие два полимино в наборе не являются совпадающими и любое добавленное в набор n-полимино будет совпадать с каким-нибудь из имеющихся.

Например, полный набор 3-полимино состоит из двух фигур,



а полный набор 4-полимино состоит из пяти фигур.



У Алёны есть полный набор n-полимино. Она считает *некрасивыми* все полимино, содержащие квадрат 2×2. По заданному n определите количество некрасивых полимино.

Input

Первая строка входа содержит одно целое число n — параметр набора полимино ($3 \le n \le 7$).

Output

Выведите одно целое число — количество некрасивых n-полимино.

Examples

стандартный ввод	стандартный вывод
3	0
4	1

Note

Так как в квадрате 2×2 4 клетки, а в 3-полимино 3 клетки, то ни одно 3-полимино не содержит квадрата 2×2.

Единственным 4-полимино, содержащим квадрат 2×2, является сам квадрат 2×2.

Problem F. Robot in the Maze

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

В некоторых комнатах лабиринта $n \times n$ расставлены указатели «север», «юг», «запад» и «восток». Остальные комнаты пусты. Робот стартует с некоторого поля и следует по указателям до тех пор, пока не выйдет из лабиринта или не попадёт на поле, в котором нет указателя. Стены в лабиринте отсуствуют, то есть робот может перейти на любое соседнее по стороне поле (или выйти из лабиринта, если он пошёл с края доски в соответствующую сторону).

Требуется выяснить, что произойдёт с роботом: зациклится ли он, покинет ли доску или придёт на какое-то свободное поле доски (в этом случае требуется указать, на какое именно).

Input

Первая строка входных данных содержит одно целое число n, задающее размер лабиринта ($1 \le n \le 100$).

Каждая из последующих n строк состоит из n символов, каждый символ описывает одну комнату. Если символ равен '`N`', двигаться надо на север, если символ равен '`E`' — на восток, если символ равен '`S`' — на юг, если символ равен '`W`' — на запад, если символ равен '`.`', то в комнате нет указателя. Первая строка является самой северной, первый столбец — самым западным.

В последней строке содержатся два целых числа R и C ($1 \le R, C \le n$) — номер (начиная с единицы) строки и столбца комнаты, из которой стартует робот.

Output

Если робот в результате движения по стрелке выходит из лабиринта, выведите 0, если он зацикливается, выведите -1, если он останавливается в какой-то комнате внутри лабиринта, выведите два целых числа — номер (начиная с 1) строки и столбца комнаты, в которой остановится робот.

Examples

стандартный ввод	стандартный вывод
5 NNNNN WEESE EN.SE WNWWE SSSSS 3 3	3 3
5 NNNNN WEESE EN.SE WNWWE SSSSS 2 2	-1
5 NNNNN WEESE EN.SE WNWWE SSSSS 3 1	-1
5 NNNNN WEESE EN.SE WNWWE SSSSS 1 3	0

Problem G. DHCP troubles

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

...Во время межпланетных сборов программистов в институте Космической Гавани на планете Латакония случилось непредвиденное: в предназначенной для участников сборов подсети обнаружилась нехватка свободных IP-адресов. Прибывший с Земли системный администратор Дмитрий просканировал логи роутера, обнаружил список адресов подключенных устройств и поручил Вам по маске подсети и списку встречающихся в логе подключения адресов выяснить, сколько адресов в данной подсети свободно. Заметим, что роутер обслуживает не только интересующую нас подсеть, а также что один и тот же адрес может упоминаться в логах многократно.

Подсеть задаётся базовым IP-адресом и маской подсети.

- Базовый IP-адрес представляет собой 32-битовое целое число и задаётся как набор из 4 целых чисел от 0 до 255 включительно, записываемых через точку (например, "`129.1.3.17`"). Первое число соответствует старшему байту ip-адреса, последнее — самому младшему. То есть записи "`129.1.3.17`" соответствует двоичное число 10000001 00000001 00000011 00010001.
- Маска подсети M представляет собой целое число от 0 до 31 и обозначает количество старших бит IP-адреса, являющихся постоянными для всех адресов в данной подсети. То есть адрес принадлежит данной подсети тогда и только тогда, когда старшие M бит адреса и маски подсети совпадают.

  Маска подсети записывается через наклонную черту сразу после базового адреса. Например, "`129.1.3.17/28`". В данном случае маска равна 28, то есть подсеть включает в себя все IP-адреса с фиксированными 28 старшими битами 10000001 00000001 00000011 0001 (то есть адреса от "`129.1.3.16`" до "`129.1.3.31`").
- При этом существуют два специальных IP-адреса, которые не могут быть назначены: наибольший адрес (где в маске нулевые биты — в нём стоят единицы) соответствует широковещательному адресу, наименьший — адресу самой подсети (где в маске нулевые биты — в нём стоят нулевые биты). В случае с подсетью "`129.1.3.17/28`" это адреса "`129.1.3.16`" и "`129.1.3.31`", тем самым до подключения первого устройства всего в этой подсети доступны 14 адресов.

Input

Первая строка входных данных. задаёт подсеть в формате, описанном в условии задачи. Все числа в IP-адресе и маска подсети не содержат в записи ведущих нулей.

Во второй строке входных данных задано одно целое число N ($1 \le N \le 100$) — количество устройств, подключенных к роутеру. Каждая из последующих N строк содержит один IP-адрес, записанный в стандартном формате — упоминание подключенного к роутеру устройства в логе. Гарантируется, что все IP-адреса корректны.

Output

Выведите одно целое число — количество свободных IP-адресов в заданной подсети. В случае, если в логах роутера обнаружены адрес или широковещательный адрес заданной подсети, то в конфигурации системы есть какая-то ошибка и надо вывести -1.

Examples

стандартный ввод	стандартный вывод
129.1.3.17/28 2 129.1.3.17 129.1.3.15	13
129.1.3.17/24 4 129.1.3.255 127.0.0.1 129.1.3.18 129.1.3.255	-1

Problem H. Array Test

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

Преподаватель Д. очень любит давать контрольные работы; вот и сегодняшний день не стал исключением. Однако на контрольной Д. дал очень странную задачу, которую Вам и требуется решить...

Напомним несколько классических определений. Пусть дан массив A из n целых чисел $[a_1, a_2, \ldots, a_n]$. Последовательность $[b_1, b_2, \ldots, b_m]$ будет называть подмассивом данного массива, если существует такое l, $1 \le l \le n - m + 1$, что $a_l = b_1, a_{l+1} = b_2, \ldots, a_{l+m-1} = b_m$.

Разные массивы даже одинаковой длины могут иметь разное количество непустых подмассивов. Например, массив $[1, 1, 1]$ имеет три различных подмассива ([1], [1, 1], [1, 1, 1]), в то время как массив $[1, 2, 3]$ - шесть ([1], [1, 2], $[1, 2, 3], [2], [2, 3], [3]$).

Массивы можно сравнивать лексикографически. Массив $C = [c_1, c_2, \ldots, c_k]$ лексикографически меньше массива $D = [d_1, d_2, \ldots, d_l]$, если выполнено одно из двух условий:

1. Массив C короче массива D и является его префиксом, то есть $k < l$, и при этом $c_1 = d_1, c_2 = d_2, \ldots, c_k = d_k$;
2. В первой позиции, в которой они отличаются, у массива C стоит меньший элемент, то есть существует такое целое положительное $p \le min(k, l)$, что $c_1 = d_1, c_2 = d_2, \ldots, c_{p-1} = d_{p-1}$ и $c_p < d_p$.

А вот и условие задачи. Пусть дан массив $A = [a_1, a_2, \ldots, a_n]$ и целое положительное число k. Упорядочим все подмассивы в лексикографическом порядке. Какой подмассив идёт в полученном упорядоченном списке k-м?

Input

В первой строке записаны числа n и k ($1 \le n \le 1\,000\,000$, $1 \le k \le 30$) — длина последовательности. Во второй строке записано n целых чисел, по модулю не превосходящих 10^6.

Output

Выведите искомый подмассив, либо -1, если такого подмассива не существует.

Examples

стандартный ввод	стандартный вывод
5 6 5 4 3 1 2	3 1 2
4 5 3 3 3 3	-1

Problem I. Favorite Points

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	6 секунд
Memory limit:	256 мебибайт

Преподаватель Д. любит давать контрольные работы. Вот и завтра он очень хочет дать контрольную работу по геометрии.

Для формулировки очередной задачи Д. необходимо задать параллелограмм, ромб, квадрат и трапецию. У Д. есть список из n любимых точек; Д. хочет выбрать каждый из четырехугольников так, чтобы все его вершины были любимыми точками, а четырехугольники - невырожденными. Однако от обилия вариантов разбегались глаза, и было непонятно, с чего начинать выбор. Д., как принято в таких случаях, просит Вас написать программу, которая по списку любимых точек найдет четыре числа — количество способов выбрать параллелограмм, ромб, квадрат и трапецию.

Д. любезно напоминает, что выпуклый четырехугольник $ABCD$ является:

- параллелограммом, если отрезок AB параллелен отрезку CD и AD параллелен BC;
- ромбом, если $|AB| = |BC| = |CD| = |DA|$;
- квадратом, если $|AB| = |BC|$ и все четыре угла четырехугольника равны;
- трапецией, если AB параллелен CD и AD НЕ параллелен BC, либо AD параллелен BC и AB НЕ параллелен CD.

Input

В первой строке записано натуральное число n, $4 \le n \le 1000$ — количество любимых точек Д. . В последующих n строках записано по два целых числа, не превосходящих по модулю 10^8 — координаты x и y точек. Гарантируется, что все точки попарно различны.

Output

Выведите искомые четыре числа. В точности следуйте формату нижеследующего примера.

Examples

стандартный ввод	стандартный вывод
4 0 0 0 1 1 0 1 1	Parallelograms: 1 Rhombuses: 1 Squares: 1 Trapezoids: 0
6 0 0 0 1 0 2 1 0 1 1 1 2	Parallelograms: 5 Rhombuses: 2 Squares: 2 Trapezoids: 4

Problem J. Thorny Graph

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

Как гласят старые малоярославские легенды, где-то далеко-далеко, где сборная России когда-то готовилась к международной олимпиаде, в одной из общеобразовательных школ живёт Граф. Ориентированность Графа, как следует из легенд, меняется от дня ко дню вместе с количеством вершин и ребер, что позволяет Графине и её отражению не скучать и играть во множество игр на Графе.

Вот уже многие дни Граф обрёл постоянную ориентированность; однако в последнее время Граф часто стал бывать раздражительным и колючим, и для Графини настали нелёгкие дни. Чтобы хоть как-то облегчить себе жизнь, Графиня решила написать программу, которая по текущему состоянию, то есть набору вершин и рёбер, сообщит Графине, является ли граф колючим. По опыту прошлых дней, Графиня заключила, что Граф является колючим, если и только если через любое его ребро проходит не более чем один простой цикл.

Напомним, что последовательность ребер (u_1, u_2), (u_2, u_3), ..., (u_{k-1}, u_k), (u_k, u_1) является простым циклом, если вершины $u_1, u_2, \ldots, u_k$ попарно различны. Простой цикл проходит через ребро e, если ребро e содержится в последовательности ребер цикла.

Петлёй в графе называется ребро (u, v), т.ч. $u = v$.

Рёбра (u_1, v_1) и (u_2, v_2) называются кратными, если $u_1 = u_2$ и $v_1 = v_2$.

Помогите Графине понять, является ли её Граф, являющийся **ориентированным** графом без петель и кратных ребер, колючим или нет.

Input

В первой строке записаны целые неотрицательные числа n и m ($1 \le n \le 500\,000$, $0 \le m \le 10^6$ - количество вершин и рёбер графа-Графа.

Далее в следующих m строках записано по паре целых чисел u, v, $1 \le u, v \le n$, $u \ne v$.

Гарантируется, что в Графе не существует петель и кратных ребер.

Output

Выведите слово "YES", если Граф является колючим, и "NO" иначе.

Examples

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

Problem K. Xor and segments

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	2.5 секунд
Memory limit:	256 мебибайт

Дан массив $a_1, a_2, \ldots, a_n$, каждое a_i равняется 0 или 1. Напишите программу, которая умеет выполнять две операции:

1. по заданным числам l и r $(1 \le l \le r \le n)$ меняет каждое из чисел $a_l, a_{l+1}, \ldots, a_r$ на противоположное (0 на 1, 1 на 0);
2. для заданных чисел l и r, $(1 \le l \le r \le n)$ рассматривает все подотрезки массива длиной от l до r, вычисляет для каждого из них сумму чисел на этом подотрезке, возвращает остаток от деления этой суммы по всем подходящим подотрезкам на 2. Формально, вычисляется величина:
$$(\sum_{i=l}^{n-l+1} \sum_{j=i+l-1}^{min(n,i+r-1)} \sum_{k=i}^{j} a_k) \ mod \ 2$$
.

Input

В первой строке записаны числа n и q $(1 \le n, q \le 250\,000)$.

Во второй строке записаны n чисел $a_1, a_2, \ldots, a_n$, каждое из этих чисел равно 0 или 1.

В следующих q строках расположены параметры запросов. В i-й из этих строк записаны параметры i-го запроса — числа t_i, l_i и r_i, где t_i равно 1 для запроса первого типа (замена чисел в массиве) и 2 для запроса второго типа (нахождение суммы), а l_i и r_i, $1 \le l_i \le r_i \le n$ есть параметры соответствующего запроса.

Output

Для каждого запроса второго типа выведите в отдельной строке единственное число — ответ на соответствующий запрос.

Example

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