Problem A. Альпинизм

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	512 мегабайт

Алексей занимается альпинизмом.

Недавно он составил план восхождений на текущий год.

План состоит в следующем. Пусть Алексей взошёл на вершину высоты X метров. Если X чётно, то Алексей даёт себе возможность отдохнуть и выбирает следующую вершину высотой вдвое меньше. Если Алексей взошёл на вершину нечётной высоты, он усложняет себе задачу и выбирает следующую вершину высотой $3X + 1$.

Высота Джомолунгмы (самой высокой вершины мира) равна 8848 метров. Если очередная вершина выше неё — план не будет выполнен. Если высота очередной (или самой первой) вершины будет в точности равна 8848 метров, то Алексею придётся взойти на Джомолунгму.

Легко увидеть, что список вершин зависит только от стартовой вершины. Назовём вершину **перспективной**, если при старте с этой вершины Алексей взойдёт на Джомолунгму до того, как план будет провален. Отсортируем список перспективных вершин по возрастанию высот (от самой низкой к самой высокой). Ваша задача — вывести i-ю вершину в этом списке, или -1, если в списке меньше i вершин.

Input

Входные данные содержат одно целое число i ($1 \le i \le 2023$).

Output

Если в списке перспективных вершин не менее i, выведите высоту i-й вершины после сортировки списка по возрастанию высот. В противном случае выведите -1.

Examples

стандартный ввод	стандартный вывод
3	8848
2023	-1

Problem B. Борьба сумо

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	512 мегабайт

Великий ёкодзуна Паджеро решил устроить сеанс одновременной борьбы в сумо!

Поединок в сумо проводится на круглой площадке — дохё. Задача сумоиста — вытолкнуть оппонента из круга. Поэтому Паджеро решил проводить матч сразу на нескольких дохё.

По правилам Федерации Сумо, центр дохё может находится только в точке с целыми координатами, а радиус дохё должен быть равен целому числу R. Сам Паджеро стоит в точке с координатами $(0, 0)$.

Чтобы ёкодзуна не проиграл поединок ещё до его начала, он должен стоять внутри или на границе соответствующего дохё. На каком максимальном количестве попарно не совпадающих дохё одновременно может провести сеанс Паджеро?

Два дохё считаются разными, если координаты их центров не совпадают.

Input

Входные данные содержат одно целое число R — радиус дохё ($1 \le R \le 10^7$).

Output

Выведите одно целое число — максимальное количество попарно различных дохё данного радиуса, на которых Паджеро может одновременно находиться (а значит, и бороться).

Examples

стандартный ввод	стандартный вывод
5	81
100	31417

Problem C. Восстанавливаем данные

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	512 мегабайт

Это задача с двойным запуском

Во время репортажа с соревнований по прыжкам в длину, проходящих в другом полушарии, вам потребовалось передать массив из N 32-битных чисел (фактически это вещественные числа, задающие результаты спортсменов, но в этой задаче это значения не имеет). К сожалению, из-за возмущений на солнце при передаче информации сегодня возможны сбои: пусть вы передаёте K чисел m_i, тогда на передающей станции случайным образом выбирается одно из этих чисел m_x и каждое из K чисел заменяется на число $m_i \oplus m_x$, где $\oplus$ — исключающее «или», после чего изменённые данные приходят получателю.

Ваша компания уже оплатила объём передаваемой информации. Путём многоуровневых переговоров вам разрешили передать на одно число больше. Требуется передать информацию и принять её после сбоя.

Ваше решение будет запущено на каждом тесте дважды — сначала на кодирование, потом на декодирование.

Input

Если вам надо передать данные, то первая строка входных данных содержит слово «`send`», вторая строка содержит целое число N — количество результатов ($1 \le N \le 10^5$), третья строка содержит N целых чисел в промежутке от 0 до $2^{32} - 1$ — сами данные.

Если вам надо принять данные, то первая строка входных данных содержит слово «`receive`», вторая строка содержит целое число K — количество переданных данных, третья строка содержит K целых чисел в промежутке от 0 до $2^{32} - 1$ — сами данные после искажения.

Output

При передаче данных выведите $N + 1$ целое число в промежутке от 0 до $2^{32} - 1$.

При приёме данных выведите $K - 1 = N$ целых чисел, которые должны в точности совпадать с числами, которые вам надо передать.

Examples

стандартный ввод	стандартный вывод
send 3 10 20 30	1 2 3 4
receive 4 3 0 1 6	10 20 30

Problem D. Главное — простота!

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	512 мегабайт

Это интерактивная задача

...На тренировке сборной математического факультета по баскетболу перед чемпионатом университета тренер несколько раз повторил следующую мысль — всегда ищите простое продолжение, незачем усложнять игру.

По дороге на матч с командой физического факультета математики придумали специальную игру, которую так и назвали — "главное — простота".

Игроки пишут на доске цифры по очереди, каждый раз приписывая новую цифру **справа** от уже существующих. Проигрывает тот, после чьего хода на доске окажется число, не являющееся простым. Например, первый ход первого игрока — это одна из цифр 2, 3, 5 и 7 (остальные однозначные числа простыми не являются). Если первый написал 7, второй может написать 1, 3 или 9 (так как 71, 73 и 79 простые), но не может, например, написать 2 (так как 72 составное) или дописать 4 спереди — цифры дописываются только в конце.

Ваша задача — сыграть с компьютером в эту игру и обыграть его. При этом компьютер делает свой ход первым.

Interaction Protocol

Взаимодействие начинает компьютер, выводя текущую цифру. Затем ваша программа выводит очередную цифру, которую надо дописать к существующему числу, чтобы число оставалось простым. В ответ на это компьютер выводит свой ход и так далее. Если в итоге после вашего хода число стало составным, решение признаётся неверным. Если компьютер не может сделать ход, он признаёт поражение и выводит -1, после чего ваша программа должна корректно завершить выполнение.

Example

стандартный ввод	стандартный вывод
2 9 -1	 3 3

Note

Не забывайте после каждого хода выводить перевод строки и сбрасывать буфер вывода вызовом функции `flush` используемого Вами языка программирования. В противном случае попытка может получить превышение лимита реального времени (Wall Time Limit Exceed).

Problem E. Документ с паролем

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	512 мегабайт

Пользователи часто забывают пароли. А если забыт пин-код от важного документа...

Компания «NeverForget» предложила новый способ вспомнить pin-код, состоящий из четырёх цифр. На экран выводится сгенерированная случайным образом формула, в которой цифры pin-кода заменены на `a`, `b`, `c` и `d`, а также значение этой формулы при подстановке правильных значений a,b,c и d.

Считается, что пользователь после этого вспомнит pin-код. Но в некоторых случаях формула может просто однозначно определить ответ, что небезопасно — доступ получит даже тот, кто вообще не знал пароля. Или же формула будет верна всегда — например, a+b+c+d-a-b-c-d равно нулю независимо от pin-кода.

Вы — стажёр в компании NeverForget. Чтобы оценивать качество подсказок, разработчики системы попросили вас написать программу, которая по заданной подсказке выдаёт количество удовлетворяющих ей pin-кодов.

Input

Первая строка входных данных содержит формулу, которую ввёл пользователь для того, чтобы запомнить pin-код.

Формула может состоять из:

- Переменных `a`,`b`,`c` и `d`, обозначающих, соответственно, первую, вторую, третью и четвёртую цифры.
- Круглых скобок '`(`' и '`)`'.
- Целых чисел от 1 до 10^4.
- Знаки арифметических операций '`+`', '`-`', '`*`'

Пробелы в записи формул отсутствуют.

Если несколько переменных записаны подряд без знаков между ними, то это обозначает десятичное число из соответствующего числа цифр. Например, при $a = 0$, $b = 1$ и $c = 2$ *abcab* обозначает число 01201 (или просто 1201, так как ведущие нули игнорируются). Такая запись называется *блоком*. Гарантируется, что длина каждого блока не превосходит 4. В остальном формула соответствует правилам арифметики (в частности, в отсутствие скобок умножение имеет приоритет над сложением и вычитанием). Унарный минус отсутствует.

Более формальное описание правила построения формул:

```
<number> :: целое число от 0 до 10000 без ведущих нулей
<var> :: a | b | c | d
<block> :: <var> | <var><var> | <var><var><var> | <var><var><var><var>
<op> :: + | - | *
<token> :: <block> | <number>
<formula> :: <token> | <sequence> <op> <sequence> | (<sequence>)
```

Формула непуста, длина её не превосходит 200. В формуле встречается не более двух знаков умножения.

Вторая строка содержит целое число N ($1 \le N \le 10^{16}$) — значение формулы.

Output

Выведите количество вариантов pin-кода, для которых значение формулы S равно числу N. Если вариантов ровно один, то во второй строке выведите сам pin-код.

Examples

стандартный ввод	стандартный вывод
a+b+c+d 36	1 9999
a+b+c+d 5	56
b+c+d 27	10
abcd+abcd 1024	1 0512
1 2	0
abcd+1083 6006	1 4923

Problem F. Есть ли шанс выиграть?

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	2 секунды
Memory limit:	512 мегабайт

До конца хоккейного матча остаётся n секунд. Счёт по-прежнему 0:0... Обе команды понимают, что надо менять тактику, и переходят к открытому хоккею.

После изменения стиля игры вероятность для домашней команды забить шайбу в каждую секунду времени равна p_h/q_h; для гостевой команды вероятность забить шайбу в каждую секунду времени равна p_v/q_v.

Команда не может забить более одной шайбы за секунду. События «домашняя команда забила» и «гостевая команда забила» являются независимыми. В частности, обе команды могут забить в одну и ту же секунду.

Найдите вероятность того, что домашняя команда одержит победу в основное время (то есть забьёт строго больше шайб, чем гости, в течение оставшихся n секунд. В рамках данной задачи процесс игры следует считать дискретным с шагом в одну секунду. Вероятность считается как отношение количества возможной последовательности событий, при которых домашняя команда выиграет в основное время, к общему количеству различных последовательностей событий (последовательности считаются разными, если хотя бы в одну секунду хотя бы одна из команд забила шайбу в одной последовательности и не забила в другой).

Input

Первая строка входных данных содержит одно целое число n — количество секунд, оставшихся до конца матча ($1 \le n \le 10^7$).

Вторая строка содержит два целых числа p_h и q_h ($0 \le p_h \le 10^9$, $1 \le q_h \le 10^9$, $p_h \le q_h$) — вероятность p_h/q_h, с которой домашняя команда забивает шайбу за секунду.

Третья строка содержит два целых числа p_v и q_v ($0 \le p_v \le 10^9$, $1 \le q_v \le 10^9$, $p_v \le q_v$) — вероятность p_v/q_v, с которой гостевая команда забивает шайбу за секунду.

Output

Выведите требуемую вероятность по модулю 998 244 353. Пусть вероятность представляется как P/Q, где P и Q — взаимно простые числа, тогда выведите число $P \cdot Q^{-1} \bmod 998\,244\,353$ (здесь $(Q^{-1}$ обозначает целое число Q_1 такое, что $Q_1 \cdot Q - 1$ делится на 998 244 353). Гарантируется, что Q и 998 244 353 взаимно просты.

Example

стандартный ввод	стандартный вывод
10 1 2 1 3	807527375

Problem G. Ёжики и яблоки

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Институт Биологического Равновесия проводит исследование о влиянии ёжиков на биосферу леса. В рамках исследования был исследован маршрут ёжиков, которые тащили яблоки между двумя фиксированными пунктами в лесу.

Маршрут был разбит на n сегментов. Исследователи установили скрытые камеры, оценивающие загруженность ёжиков на этом сегменте. Загруженность оценивается целым числом от 1 до m, причём на первом и на n-м сегменте загруженность оказалась минимальной (то есть равной 1). Также оказалось, что любое значение загруженности m встретилось среди результатов хотя бы раз.

По вычисленной последовательности построили матрицу переходов $K_{x,y}$ размера $m \times m$, определяемую следующим образом: $K_{x,y}$ — количество таких значений i от 1 до $n-1$, что для i-го сегмента загруженность ёжиков равна x, а для $i+1$-й — y.

При дальнейшей обработке использовалась только матрица, исходные же данные не сохранились. Сейчас разработчики системы хотят понять, сколько различных наборов исходных данных могут соответствовать заданной матрице переходов.

Input

Первая строка входных данных содержит одно целое число m — количество возможных значения загруженности ($2 \le m \le 400$).

Последующие m строк, каждая из которых содержит по m чисел, задают матрицу K. Элементы матрицы являются целыми неотрицательными числами, не превосходящими 500, причём хотя бы один элемент матрицы не равен нулю.

Output

Выведите одно число — остаток деления от количества вариантов исходных наборов данных на 998 244 353.

Example

standard input	standard output
4 2 0 0 2 0 3 0 1 2 1 0 0 0 0 3 1	36

Problem H. Женский баскетбол

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	4 секунды
Memory limit:	512 мегабайт

Методы анализа статистики в современном женском баскетболе сильно спрогрессировали. Перед началом нового сезона аналитики команд оперируют огромным количеством параметров об огромном количестве перспективных (и не очень) участниц школьных и студенческих соревнований.

И оперируют аналитики уже не только с последовательностями, но и с последовательностями последовательностей. И это же самое должен уметь делать и софт.

Вы реализуете для аналитической программы следующий модуль: есть N последовательностей параметров A_1, A_2, …, A_N, каждая из которых состоит из N элементов. Обозначим за $A_{i,j}$ j-й элемент последовательности A_i.

Изначально $A_{i,1} = a_1$, $A_{i,2} = a_2$, …, $A_{i,N} = a_n$ для всех N последовательностей A_i.

Ваша задача — выполнять следующие запросы:

- 1 x y — заменить последовательность A_y последовательностью A_x **целиком**.
- 2 x s y t — поменять местами значение s-го элемента $A_{x,s}$ последовательности A_x и значение t-го элемента $A_{y,t}$ последовательности A_y.
- 3 x s f — вывести сумму элементов последовательности A_x, начиная с s-го и заканчивая f-м, соответственно.

Input

Первая строка входных данных содержит два целых числа N и Q ($2 \le N \le 2 \cdot 10^5$, $1 \le Q \le 2 \cdot 10^5$) — количество последовательностей и количество запросов, соответственно.

Вторая строка содержит N целых чисел a_i ($1 \le a_i \le 10^9$) — первоначальные значения i-х элементов **каждой** из последовательностей A_j.

Далее следуют Q строк, задающих запросы, по одному запросу на строку.

Запрос типа 1 задаётся тремя целыми числами: 1, x, y ($1 \le x, y \le N$) и обозначает, что мы заменяем последовательность A_y последовательностью A_x (то есть выполняем присванивание $a_{i,y} = a_{i,x}$ для всех i между 1 и N включительно).

Запрос типа 2 задаётся пятью целыми числами: 2, x, s, y, t ($1 \le x, y, s, t \le N$) и обозначает, что мы меняем местами значения $a_{x,s}$ и $a_{y,t}$.

Запрос типа 3 задаётся тремя целыми числами: 3, x, s, f ($1 \le x, s \le f \le N$), в качестве ответа на этот запрос выведите сумму элементов последовательности A_x между элементами $a_{x,s}$ и $a_{x,f}$ включительно.

Output

Для каждого запроса типа 3 выведите ответ в новой строке.

Example

стандартный ввод	стандартный вывод
4 4 108 3 600 6 3 1 2 3 2 3 3 1 2 1 1 3 3 3 2 3	603 1200