Problem A. Арифметика: сложение

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	2 секунды
Memory limit:	1024 мебибайта

Вычислите сумму двух целых чисел.

Input

Единственная строка входных данных содержит два целых числа A и B, не превосходящих по абсолютной величине 1 000.

Output

Выведите результат вычисления $A + B$.

Examples

стандартный ввод	стандартный вывод
2 3	5
17 -18	-1

Problem B. Будем делить

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	2 секунды
Memory limit:	1024 мебибайт

Разделите одно целое число на другое и выведите вещественный результат деления.

Input

Входной файл состоит из двух целых чисел A и B, не превосходящих по абсолютной величине 100.

Output

Ваша программа должна вывести одно вещественное число — результат деления A на B. Ответ будет принят, если будет отличаться от правильного не более, чем на 0.01.

Examples

стандартный ввод	стандартный вывод
4 2	2.0
2 3	0.667

Problem C. Выставляем баллы

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	2 секунды
Memory limit:	1024 мебибайта

Если вы решили эту задачу, вы можете попробовать свои силы в дивизионе D

Это интерактивная задача: ваша программа будет обмениваться сообщениями с программой жюри. При этом очередная часть ввода от программы жюри поступает только после того, как получен ответ от вашей программы. Протокол обмена обычно описывается в разделе Interaction (взаимодействие).

В соревновании по решению оптимизационной задачи методами машинного обучения в тестирующей системе Kaggjudge попытки, отправленные по задаче, оцениваются целыми баллами от 0 до 10^6 включительно, результат участника на соревновании равен максимальному баллу, полученному за одну из его попыток.

При просмотре решений участников постановщики задачи обнаружили, что в некоторых случаях решения используют средства, не входящие в список разрешённых. После чего проверяющая программа была исправлена таким образом, что использование этих средств в решении приводит к выставлению оценки в 0 баллов.

Проверка одного решения задачи, предложенной на текущем соревновании, занимает очень большое время (и использует все доступные вычислительные ресурсы), так что хочется определить результат каждого участника после исправления проверяющей программы за наименьшее количество перепроверок.

Вам даётся список попыток, сделанных на соревновании одним участником, и набранные соответствующими решениями баллы. Все ненулевые баллы попарно различны. За один запрос вы можете отправить одну посылку на перепроверку и узнать результат перепроверки: 1, если существующий балл сохраняется, и 0, если посылка набирает 0 баллов.

Ваша задача — достоверно определить результат участника за наименьшее количество перепроверок.

Interaction Protocol

Взаимодействие начинает программа жюри, которая выводит целое число N ($1 \le N \le 10$) — количество отправленных участником решений, а в следующей строке — N целых чисел p_i от 0 до 10^6. i-е из этих чисел задаёт текущий балл, набранный соответствующим решением за задачу. При этом все p_i, не равные нулю, являются попарно различными.

Далее программа жюри сообщает целое число T ($1 \le T \le 2^N$) — количество сценариев.

Сценарии являются независимыми (то есть перед началом сценария балл за i-е решение равен p_i). **Перед** началом каждого сценария программа жюри определяет, какие попытки при перетестировании сохранят результат, а какие получат 0 баллов (то есть эти результаты не зависят от действий вашей программы).

Взаимодействие в рамках каждого сценария начинает ваша программа, выводя запрос на перетестирование в виде `retest` i, где $1 \le i \le N$ — номер решения, которое надо перетестировать. В ответ на этот запрос программа жюри выводит 0, если решение получает 0 баллов, или 1, если балл за решение сохраняется. Как только результат участника гарантированно станет равным результату по задаче после внесения исправления, программе жюри сообщается об этом командой `done`.

Можно доказать, что для каждого сценария существует оптимальное число запросов. Если результат неправильный (или число запросов было больше оптимального в данном сценарии), выполнение прерывается с вердиктом Wrong Answer. В противном случае программа жюри переходит к взаимодействию по следующему сценарию (если он есть).

Example

стандартный ввод	стандартный вывод
3 10 20 30 2 0 0 0 0 0 1	 retest 1 retest 2 retest 3 done retest 2 retest 1 retest 3 done

Note

Пример дан только для понимания формата: в сценариях из примера гарантируется только корректный итоговый балл за решение, но не оптимальность.

После вывода каждого запроса и команды `done` не забывайте выводить перевод строки и очищать буфер вывода, иначе решение получит вердикт «`Wall Time Limit Exceeded`». Очистить буфер можно, например, вызовом `fflush (stdout)` в C или C++, `System.out.flush ()` в Java, или `sys.stdout.flush ()` в Python.

Problem D. Гиперинфляция

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	2 секунды
Memory limit:	1024 мебибайта

Если вы решили эту задачу, вам стоит задуматься о выборе дивизиона D или выше

Резкий рост инфляции в Берляндии привёл к падению спроса на бриллианты. Так что неудивительно, что в берляндский супермаркет бриллиантов пришёл новый директор по маркетингу.

Первым его решением стало изменение ценовой политики на более привлекательную. До этого все цены в супермаркете выражались целым числом бурлей. Новый директор решил перенять проверенные передовые приёмы и предложил уменьшить каждую цену на 1 покейку (1 покейка = 1/100 бурля).

Ценники в супермаркете набираются из пластиковых цифр, так что для реализации распоряжения директора требуется докупить какое-то количество «девяток». По заданному текущему списку цен на товары супермаркета выясните, сколько дополнительных девяток потребуется для записи цен после «беспрецедентного снижения». Заметим, что в используемом в магазине шрифте цифру 9 нельзя получить, перевернув цифру 6.

Input

Первая строка входных данных содержит одно целое число N ($1 \le N \le 1000$) — количество позиций в супермаркете. Каждая из последующих строк содержит одно целое число p_i ($1 \le p_i < 10^{1000}$) — исходная цена очередной позиции в бурлях.

Output

Выведите одно целое число — количество дополнительных девяток в списке цен после того, как предложение директора по маркетингу будет реализовано.

Example

стандартный ввод	стандартный вывод
3 15 10 2023	7

Problem E. Досадный сбой

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Если вы решили эту задачу, то вам стоит участвовать как минимум в дивизионе C, и вы можете попробовать свои силы в отборе дивизиона AB

Это задача с двойным запуском.

У Вас есть массив из $n \leq 1000$ беззнаковых 64-битных целых чисел. Вы собираетесь быстро переслать его на другой компьютер. Для этого вы параллельно отправляете каждое число через Интернет и потом снова собираете массив.

Однако возникла непредвиденная проблема. Как известно, «сетевой» порядок байт в многобайтовом числе отличается от используемого в современных компьютерах: если в современном компьютере байты записываются от младшего к старшему (little-endian), то при сетевом порядке байты записываются от старшего к младшему (big-endian). При преобразовании каждое число записывается как последовательность из 8 байт, и байты записываются в обратном порядке. И в некоторых случаях из-за сбоя на серверах обратное преобразование сделано не было...

То есть каждый элемент массива может прийти либо в обычном little-endian порядке, либо в сетевом big-endian. Порядок элементов массива сохранён. При передаче вы можете использовать не более 1024 64-битных чисел (то есть не более 8 **кибибайт**). Ваша задача — передать исходный массив и восстановить его после возможных искажений.

Input

Если требуется передать массив, то в первой строке записано слово «`encode`», во второй — целое число n ($1 \leq n \leq 1000$), а в третьей — n целых чисел в интервале от 0 до $2^{64} - 1$.

Если требуется получить массив, то в первой строке записано слово «`decode`», во второй — целое число k ($k \leq 1024$), а в третьей — k целых чисел в интервале от 0 до $2^{64} - 1$. Гарантируется, что числа идут в том же порядке, в котором они были переданы, и что любое из чисел либо передаётся неизменным, либо имеет перевёрнутый порядок байтов (в соответствии с условием задачи).

Output

В случае передачи массива в первой строке выведите одно целое число $k \leq 1024$: количество передаваемых чисел. Во второй строке выведите k целых чисел в интервале от 0 до $2^{64} - 1$.

В случае получения массива выведите n целых чисел — исходный массив.

Examples

standard input	standard output
encode 3 15 10 2023	6 15 15 10 10 2023 2023
decode 6 15 15 10 720575940379279360 1664727454759832780 8 2023	15 10 2023

Note

В нижнем примере — в случае получения массива — все шесть чисел будут даны в одной строке. Дополнительный перевод строки добавлен только для удобства чтения.

На каждом тесте ваша программа будет запущена два раза: в первый раз на передачу массива, а во второй раз на получение, причём в качестве входных данных будет передан вывод первого запуска с возможными искажениями. Тест считается пройденным, если исходный массив восстановлен корректно.