

## Задача А. Абстракционист и треугольники

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 1 секунда  
Ограничение по памяти: 1024 мегабайта

*Это задача с двойным запуском*

Требуется закодировать треугольник ненулевой площади с целыми сторонами и периметром  $p$ , никакие две стороны которого не равны, треугольником ненулевой площади и периметром  $p - 6$ , и затем по закодированному треугольнику восстановить исходный.

### Формат входных данных

Первая строка входных файлов содержит режим работы: `encode`, если требуется закодировать треугольники или `decode`, если требуется восстановить исходные. Вторая строка содержит одно целое число  $t$  ( $1 \leq t \leq 25\,000$ ) — количество тестовых примеров.

Далее следуют тестовые примеры по одному на строку.

В режиме кодирования каждый тестовый пример содержит три целых числа  $a$ ,  $b$  и  $c$  — стороны исходного треугольника. Гарантируется, что  $3 \leq a + b + c \leq 1000$ , что  $a$ ,  $b$  и  $c$  попарно не равны и что треугольник со сторонами  $a$ ,  $b$  и  $c$  существует и имеет ненулевую площадь.

В режиме декодирования каждый тестовый пример содержит три целых числа  $a'$ ,  $b'$  и  $c'$  — переставленные в каком-то порядке стороны, выведенные вашей программой в режиме кодирования.

### Формат выходных данных

В режиме кодирования выведите для каждого тестового примера в произвольном порядке три целых числа  $a'$ ,  $b'$ ,  $c'$ , такие, что треугольник со сторонами  $a'$ ,  $b'$ ,  $c'$  существует и имеет ненулевую площадь, и  $a' + b' + c' = a + b + c - 6$ .

В режиме декодирования для каждого тестового примера выведите три стороны исходного треугольника. Порядок вывода сторон внутри одного тестового примера произвольный.

### Примеры

| стандартный ввод | стандартный вывод |
|------------------|-------------------|
| encode           | 18 18 13          |
| 2                | 25 30 49          |
| 20 24 11         |                   |
| 25 36 49         |                   |
| decode           | 11 20 24          |
| 2                | 25 36 49          |
| 18 13 18         |                   |
| 25 49 30         |                   |

## Задача В. Четные и нечетные сочетания

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 2 секунды         |
| Ограничение по памяти:  | 512 мегабайт      |

**Это задача с двойным запуском.**

*Сочетанием из  $n$  элементов по  $k$*  будем называть  $k$ -элементное подмножество  $n$ -элементного множества  $\{1, 2, \dots, n\}$ . Чтобы записать сочетание, перечислим его элементы в порядке возрастания. Например, сочетания из 3 элементов по 2 выглядят так:  $\{1, 2\}$ ,  $\{1, 3\}$ ,  $\{2, 3\}$ .

Будем называть сочетание *четным*, если количество элементов в нём — четное число, и *нечетным* в противном случае. Зафиксируем  $n > 0$  и рассмотрим два множества:  $A_n$ , множество всех чётных сочетаний из  $n$  элементов, и  $B_n$ , множество всех нечетных сочетаний из  $n$  элементов. Можно доказать, что  $A_n$  и  $B_n$  содержат одинаковое количество сочетаний.

Для каждого  $n = 1, 2, \dots, 50$  задача такова. Постройте любую биекцию (взаимно однозначное соответствие) между множествами  $A_n$  и  $B_n$ . После этого по данному элементу одного из этих множеств выводите соответствующий ему элемент другого множества.

### Формат входных данных

При первом запуске в первой строке записано целое число  $t$  — количество тестовых случаев ( $1 \leq t \leq 1000$ ). Далее следуют их описания.

Каждый тестовый случай задает сочетание и занимает две строки.

В первой из них записаны через пробел два целых числа  $n$  и  $k$  ( $1 \leq n \leq 50$ ,  $0 \leq k \leq n$ ).

Во второй записаны через пробел  $k$  целых чисел  $a_1, a_2, \dots, a_k$  — элементы сочетания ( $1 \leq a_1 < a_2 < \dots < a_k \leq n$ ). Если  $k = 0$ , то вторая строка пуста.

При втором запуске формат ввода точно такой же, как при первом запуске. Но в каждом тестовом случае дано не исходное сочетание, а то, которое было выведено при первом запуске.

### Формат выходных данных

При первом запуске в ответ на каждый тестовый случай выведите сочетание из другого множества, соответствующее заданному. Формат сочетания в выводе — точно такой же, как во вводе. У выведенного сочетания  $n$  должно совпадать с заданным, а  $k$  должно иметь другую четность. Других ограничений на выбор соответствия нет.

При втором запуске в ответ на каждый тестовый случай, как и при первом запуске, выведите сочетание из другого множества, соответствующее заданному. Формат сочетания в выводе — точно такой же, как во вводе. Поскольку соответствие должно быть взаимно однозначным, выведенное при втором запуске сочетание должно совпадать с тем, которое было дано при первом запуске. Это и будет проверять программа жюри.

### Протокол взаимодействия

Для проверки того, что вы действительно построили биекцию, в этой задаче ваше решение будет запущено на каждом тесте два раза. В конце каждой строки входных данных следует символ перевода строки.

### Пример

Обратите внимание, что в примерах приведены конкретные варианты вывода и ввода при втором запуске, в зависимости от вашего вывода на первом запуске, ввод на втором запуске может быть другим.

| стандартный ввод   | стандартный вывод   |
|--|---|
| 6<br>3 0<br><br>2 1<br>1<br>3 3<br>1 2 3<br>3 1<br>1<br>3 1<br>2<br>3 1<br>3         | 3 3<br>1 2 3<br>2 2<br>1 2<br>3 0<br><br>3 2<br>2 3<br>3 2<br>1 3<br>3 2<br>1 2 |
| 6<br>3 3<br>1 2 3<br>2 2<br>1 2<br>3 0<br><br>3 2<br>2 3<br>3 2<br>1 3<br>3 2<br>1 2 | 3 0<br><br>2 1<br>1<br>3 3<br>1 2 3<br>3 1<br>1<br>3 1<br>2<br>3 1<br>3         |

## Задача С. Тожжество

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 2 секунды         |
| Ограничение по памяти:  | 512 мегабайт      |

Строкой Фибоначчи длины  $n$  называется последовательность из  $n$  нулей и единиц, в которой нет двух единиц подряд. Число различных строк Фибоначчи длины  $n$  будем записывать как  $F_n$ . и называть числом Фибоначчи. Известно, что  $F_n = F_{n-1} + F_{n-2}$ . Обратите внимание: последовательность начинается с  $F_0 = 1$  и  $F_1 = 2$ , чем отличается от классической нумерации чисел Фибоначчи! Например, при  $n = 3$  существует пять различных строк Фибоначчи: это «000», «001», «010», «100» и «101».

Сочетанием из  $m$  элементов по  $k$  называется последовательность целых чисел  $a_1, a_2, \dots, a_k$ , в которой все числа лежат в пределах от 1 до  $m$  и записаны в порядке строгого возрастания. Число различных сочетаний из  $m$  элементов по  $k$  записывается как  $C_m^k$ . Известно, что  $C_m^k = C_{m-1}^{k-1} + C_{m-1}^k$ , при этом  $C_0^0 = 1$ , а кроме того,  $C_m^k = 0$  при  $k > m$ . Например, при  $m = 4$  и  $k = 2$  существует шесть различных сочетаний: это (1, 2), (1, 3), (1, 4), (2, 3), (2, 4) и (3, 4).

Недавно Ада прочитала в книге следующее тождество, связывающее эти объекты:

$$F_n = C_{n+1}^0 + C_n^1 + C_{n-1}^2 + C_{n-2}^3 + \dots$$

Суммирование ведётся, пока верхний индекс числа сочетаний не превосходит нижнего: далее все значения равны нулю. Например, при  $n = 3$  тождество принимает вид

$$F_3 = C_4^0 + C_3^1 + C_2^2.$$

Ада хочет показать тождество своему другу Чарли. Она придумала, как доказать тождество по индукции. Но Чарли не любит формальные рассуждения: ему больше по вкусу конкретные примеры и наглядные доказательства. Чарли сможет по достоинству оценить тождество, если построить биекцию: взаимно однозначное соответствие одних объектов другим. Зафиксируем число  $n$ . Каждой строке Фибоначчи длины  $n$  следует сопоставить одно из сочетаний, перечисленных в правой части равенства: либо сочетание из  $n + 1$  элемента по 0, либо из  $n$  элементов по 1, либо из  $n - 1$  элемента по 2, и так далее.

Помогите Аде построить биекцию для Чарли. Напишите программу, которая любой строке Фибоначчи длины  $n$  сопоставит либо сочетание из  $n + 1$  элемента по 0, либо из  $n$  элементов по 1, либо из  $n - 1$  элемента по 2, и так далее. Затем, получив параметры сочетания и само сочетание, программа должна восстановить исходную строку Фибоначчи.

### Протокол взаимодействия

В этой задаче ваше решение будет запущено на каждом тесте два раза. При вводе и выводе числа в строках отделяются друг от друга пробелами. В конце каждой строки входных данных следует символ перевода строки.

При первом запуске решение по строке Фибоначчи строит сочетание. В первой строке записано слово «first». Вторая строка содержит целое число  $n$  — длину заданной строки Фибоначчи ( $1 \leq n \leq 300\,000$ ). В третьей строке записана сама строка Фибоначчи —  $n$  двоичных цифр, среди которых нет двух единиц подряд.

В первой строке выведите два целых числа  $m$  и  $k$ : параметры сочетания (должно быть выполнено  $m + k = n + 1$ ). Во второй строке выведите само сочетание:  $k$  различных целых чисел от 1 до  $m$ , перечисленных в порядке возрастания. Если  $k = 0$ , вторую строку (пустую в этом случае) можно как вывести, так и пропустить. Как именно сочетание зависит от заданной строки Фибоначчи — решать вам.

При втором запуске решение по параметрам сочетания и самому сочетанию восстанавливает строку Фибоначчи. В первой строке записано слово «second». Вторая строка содержит целые числа  $m$  и  $k$  — параметры сочетания. В третьей строке задано само сочетание —  $k$  различных целых чисел

от 1 до  $m$ , перечисленных через пробел в порядке возрастания. Все эти числа — ровно те, которые решение вывело при первом запуске. Если  $k = 0$ , третья строка входных данных пуста.

В первой строке выведите  $n$  — длину исходной строки Фибоначчи (должно быть выполнено  $m + k = n + 1$ ). Во второй строке выведите строку Фибоначчи длины  $n$ , совпадающую с исходной.

**После вывода не забудьте выполнить операцию flush! В противном случае вы можете получить произвольный вердикт.**

## Система оценки

Баллы за подзадачу начисляются только в том случае, если все тесты этой подзадачи и всех предыдущих подзадач успешно пройдены.

| Подзадача | Баллы | Ограничения                    | Информация о проверке |
|-----------|-------|--------------------------------|-----------------------|
| 1         | 40    | $n \leq 10$                    | До первой ошибки      |
| 2         | 35    | $n \leq 40$                    | До первой ошибки      |
| 3         | 15    | $n \leq 1000$                  | До первой ошибки      |
| 4         | 10    | Дополнительных ограничений нет | До первой ошибки      |

## Примеры

Обратите внимание, что в примерах приведены конкретные варианты вывода и ввода при втором запуске, в зависимости от вашего вывода на первом запуске, ввод на втором запуске может быть другим.

| стандартный ввод     | стандартный вывод |
|----------------------|-------------------|
| first<br>3<br>101    | 2 2<br>1 2        |
| second<br>2 2<br>1 2 | 3<br>101          |
| first<br>3<br>000    | 4 0               |
| second<br>4 0        | 3<br>000          |

## Задача D. Прерванная сортировка

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 2 секунды         |
| Ограничение по памяти:  | 512 мегабайт      |

**Это задача с двойным запуском.**

Петя — это робот, основной задачей которого является сортировка массивов. В его памяти хранится массив длины  $n$  (от 1 до 100), элементы которого являются попарно различными целыми числами. Элементы массива пронумерованы числами от 1 до  $n$  слева направо.

Нина является оператором данного робота. Нина хочет отсортировать массив, хранящийся в памяти робота. Массив называется отсортированным, если для любых двух номеров элементов  $i$  и  $j$  ( $i < j$ )  $i$ -й элемент массива меньше, чем  $j$ -й элемент массива.

Единственная команда, доступная для выполнения роботом, это сравнение двух элементов на позициях  $i$  и  $j$ . Если элемент на левой позиции (ей может оказаться как  $i$ , так и  $j$ ) больше, чем элемент на правой позиции, Петя меняет данные элементы местами и выводит число 1 на экран. В противном случае Петя ничего не делает и выводит число 0 на экран.

К сожалению, аккумулятор Пети поврежден, поэтому иногда робот выключается. Данное событие выглядит следующим образом: вместо исполнения очередной команды Петя выводит число  $-1$  на экран и игнорирует все последующие команды.

Для того, чтобы включить Петю, Нина разбирает его, заменяет аккумулятор и собирает снова. При этом массив, хранящийся в памяти, не меняется. К сожалению, во время ремонта Нина забывает, какие команды робот выполнял ранее. После ремонта Петя работает до тех пор, пока его аккумулятор не разрядится. После этого он снова выключается.

Аккумулятора Пети достаточно, чтобы выполнить 1 500 команд. Во время данного процесса Петя выключается ровно дважды: после выполнения  $x$ -й команды и после выполнения 1 500-й команды ( $0 < x < 1 500$ , значение  $x$  не известно Нине). Зная это, помогите Нине выдать роботу необходимые команды таким образом, чтобы после его второго выключения массив оказался отсортированным.

### Протокол взаимодействия

В этой задаче ваше решение будет запущено на каждом тесте два раза. Ваше решение играет за Нину, а программа жюри — за Петю.

Ниже описан протокол взаимодействия с программой жюри. Протокол одинаков для обоих запусков.

Первая строка сходных данных содержит целое число  $n$  ( $1 \leq n \leq 100$ ) — размер массива. После этого ваше решение должно выводить команды, а программа жюри будет выполнять их и выдавать результаты выполнения команд.

Для того, чтобы выполнить команду «сравнить элементы на позициях  $i$  и  $j$ », выведите строку вида « $i j$ », где  $1 \leq i, j \leq n$ . В качестве результата выполнения команды вы получите одно целое число:

- Число 1 означает, что Петя поменял элементы на позициях  $i$  и  $j$  местами, потому что левое число было больше, чем правое.
- Число 0 означает, что Петя не менял элементы местами, потому что левое число было не больше, чем правое (то есть либо левое число было меньше, чем правое, либо  $i = j$ ).
- Число  $-1$  означает, что Петя выключился вместо выполнения команды.

В последнем случае ваше решение должно немедленно завершить свою работу. В остальных случаях вы можете запросить выполнение очередной команды.

Если вы хотите прекратить взаимодействие с Петей до того, как он выключится, вместо команды выведите строку « $-1 -1$ ». После этого ваше решение должно немедленно завершить свою работу.

В каждом тесте массив зафиксирован заранее перед первым запуском. Массив перед вторым запуском находится в состоянии, в котором он был в конце первого запуска. Также в каждом тесте

число команд  $x$ , после которых Петя выключится в первый раз ( $0 < x < 1500$ ) зафиксировано заранее. Во время второго запуска робот выключится после  $1500 - x$  команд, даже в том случае, если во время первого запуска взаимодействие с Петей было завершено до его выключения.

### Пример

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

## Задача Е. Сообщение из шума

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 2 секунды         |
| Ограничение по памяти:  | 512 мегабайт      |

**Это задача с двойным запуском.**

Алиса хочет передать Еве по *числовому проводу* сообщение — одно английское слово.

К сожалению, сейчас числовой провод передаёт только шум: случайные целые числа от 0 до  $10^9 - 1$  включительно. Алисе известна последовательность из следующих 10 000 чисел, которые будут переданы.

К счастью, у Алисы есть суперспособность: стереть из этой последовательности любое количество элементов на любых позициях. Порядок оставшихся чисел не поменяется.

К сожалению, после этого примерно половина чисел потеряется при передаче: каждое передаваемое число с вероятностью  $\frac{1}{2}$  исчезнет. Порядок оставшихся чисел опять не поменяется.

Как должны действовать Алиса и Ева, чтобы передать заданное слово?

### Формат входных данных

В этой задаче ваше решение будет запущено на каждом тесте два раза. При вводе и выводе числа в одной строке отделяются друг от друга пробелами. В конце каждой строки входных данных следует символ перевода строки.

### Формат выходных данных

При первом запуске решение действует за Алису. В первой строке записано имя «Alisa». Вторая строка содержит одно слово из английского словаря, имеющее длину от 2 до 15 букв и состоящее из маленьких букв. В третьей строке записано целое число  $n$  — размер последовательности (в этой задаче  $n$  всегда равно 10 000). Четвертая строка содержит  $n$  чисел от 0 до  $10^9 - 1$  включительно — исходную последовательность. Числа выбраны заранее генератором псевдослучайных чисел, все целые числа из диапазона равновероятны.

При втором запуске решение действует за Еву. В первой строке записано имя «Eva». Во второй строке записано целое число  $k$  — количество чисел, оставшихся в последовательности. Третья строка содержит  $k$  чисел от 0 до  $10^9 - 1$  включительно — то, что осталось от последовательности: каждое число, которое Алиса решила оставить в первом запуске, с вероятностью  $\frac{1}{2}$  оказалось здесь и с вероятностью  $\frac{1}{2}$  исчезло. Как именно исчезают числа из последовательности — зафиксировано заранее в каждом тесте, то есть, если решения ведут себя одинаково при первом запуске, то они получат одинаковые последовательности при втором запуске.

### Протокол взаимодействия

При первом запуске вывести следует те числа, которые Алиса решила **оставить** в последовательности. В первой строке выведите целое число  $m$  — количество оставшихся чисел. Во второй строке выведите сами эти числа в том же порядке, в котором они следуют в исходной последовательности.

При втором запуске выведите одно английское слово — то, которое Алиса должна была передать Еве.

### Пример

Обратите внимание, что в примерах приведены конкретные варианты вывода и ввода при втором запуске, в зависимости от вашего вывода на первом запуске, ввод на втором запуске может быть другим.

Для краткости последовательности в примере показаны не полностью. Полную версию примера можно найти в `samples.zip`.

| стандартный ввод  | стандартный вывод   |
|---|---|
| Alisa<br>spark<br>10000<br>833080662 16249270 933346436<br>811379468 <...> 13286897 459644281 | 3900<br>933346436 811379468 877083772<br>408973036 <...> 583178591 13286897 |
| Eva<br>1955<br>811379468 408973036 585189166<br>111199534 <...> 226510051 829146141           | spark   |

## Задача F. В поисках истины

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

### Это интерактивная задача.

После вывода очередной строки не забывайте очищать буфер потока вывода после каждого запроса. Для этого можно, например, воспользоваться командами `fflush(stdout)`, `cout.flush()`, либо выполнять перевод строки при помощи `endl` в C++, `system.out.flush()` в Java, `flush(output)` в Pascal или `sys.stdout.flush()` в Python.

Сарком управлял совет Великих сквайров. В совете состояло пятеро саркитов, и самый богатый и влиятельный из них, сквайр Файф, взял на себя ответственность разобраться в истории космоаналитика. Волей случая к нему попал и Рик, который оказался космоаналитиком, а Селим Джунц и Людиган Эбл из посольства Трантора были готовы с ним сотрудничать. Сквайра интересовало одно — кто же мог совершить столь ужасное преступление?

Еще раньше Великому сквайру поступали анонимные письма, в которых сообщалось, что Флорина должна погибнуть. Шантажист требовал отдать значительную долю кыртовых полей Файфа за эту информацию. Сквайр подозревал, что если преступник смог похитить космоаналитика, прятать его целый год от властей и при этом шантажировать самого главного человека на всем Сарке, он тоже должен быть Великим сквайром. И больше всего подозрений вызывал у него сквайр Стин.

Тут неожиданно Рик вспомнил, что во время разговора с похитителем, тот сообщил размер его владений на Флорине —  $k$  квадратных километров. В архиве хранятся данные о площади земель  $s_i$ , контролируемых сквайром Стином, в  $n$  моментов времени. Файфу известно, что все  $s_i$  различны, а так же что до некоторого момента эти площади увеличивались, а потом начали убывать. Более формально, существует такое  $1 \leq t \leq n$ , что для любого  $1 < i \leq t$   $s_{i-1} < s_i$  и для любого  $t < j \leq n$   $s_{j-1} > s_j$ . Чтобы подтвердить свою правоту, Великому сквайру нужно узнать, в какой момент времени площадь владений Стиня в точности равнялась  $k$ , и он просит вас помочь ему. Вы можете по моменту времени  $i$  узнать размер владений сквайра Стиня  $s_i$ . Небольшая сложность состоит в том, что времени у Файфа мало, а направление запроса в архив происходит достаточно долго. Поэтому у вас есть возможность задать не более 80 таких запросов. Сквайр не сомневается в своем успехе, и гарантирует вам, что искомое  $s_i$  существует.

### Протокол взаимодействия

Изначально вам заданы два числа  $n, k$  — количество записей о владениях Стиня в архиве и значение площади, которое интересует Файфа ( $2 \leq n \leq 2 \cdot 10^5, 0 \leq k \leq 10^9$ ). Далее ваша программа может делать запросы вида “?  $i$ ”, в качестве ответа на которые программа жюри будет выводить значения  $s_i$ . Все  $s_i$  — целые числа,  $0 \leq s_i \leq 10^9$ . Записи в архиве нумеруются с 1. Когда ваша программа будет готова дать ответ, она должна вывести “!  $i$ ”, где  $s_i = k$ , и завершиться. Если ваша программа сделает больше 80 запросов первого типа, решение получит вердикт “Wrong answer”. Если программа не завершится после запроса второго типа или ответ на запрос второго типа будет неверным, решение также получит вердикт “Wrong answer”.

### Пример

| стандартный ввод | стандартный вывод |
|------------------|-------------------|
| 5 3              | ? 5               |
| 2                | ? 4               |
| 8                | ? 3               |
| 10               | ? 1               |
| 1                | ? 2               |
| 3                | ! 2               |

### Замечание

В данном примере записи о владениях выглядели так: 1, 3, 10, 8, 2.

## Задача G. Угадай массив

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 512 мегабайт

Это интерактивная задача. Ваша программа будет взаимодействовать с программой жюри, используя стандартный ввод и вывод.

Алиса и Боб решили сыграть в игру под названием «Угадай массив». Правила игры очень простые: Алиса загадала массив целых чисел размера  $n$ , а Бобу нужно угадать этот массив, сделав не больше  $n$  запросов о сумме на отрезке.

За один ход Боб может сделать к Алисе запрос одного из двух типов:

- «? l r» — узнать сумму чисел на отрезке массива с  $l$ -го по  $r$ -й элемент включительно;
- «!» — сообщить Алисе, что он готов дать ответ. После этого запроса Алиса ожидает от Боба  $n$  целых чисел — загаданный массив.

На каждый запрос первого типа, Алиса говорит Бобу сумму чисел на запрошенном отрезке. Но чтобы отгадывать было сложнее, после каждого запроса Алиса делает один отрезок *запрещенным*. В дальнейших запросах Боб не может запрашивать сумму чисел на запрещенных отрезках.

Помогите Бобу угадать загаданный массив, сделав не более  $n$  запросов первого типа.

### Протокол взаимодействия

В первой строке ввода дано целое число  $n$  — размер загаданного массива ( $1 \leq n \leq 10^4$ ). Гарантируется, что все числа в массиве по модулю не превосходят  $10^9$ .

Далее запускается протокол взаимодействия с программой жюри — интерактором.

Интерактор ожидает от вашей программы запросы двух типов: «? l r» или «!», где  $l, r$  — целые числа — границы отрезка, на котором вы хотите узнать сумму ( $1 \leq l \leq r \leq n$ ). После каждого запроса должен следовать перевод строки. При несоблюдении вашей программой формата запросов, ваше решение может получить произвольный вердикт (отличный от ОК).

После запроса первого типа необходимо считать со стандартного ввода три целых числа  $s, l_b$  и  $r_b$  — сумму чисел на запрошенном отрезке  $s$  и границы нового *запрещенного* отрезка  $[l_b, r_b]$ , на котором запрашивать сумму больше нельзя ( $|s| \leq 10^{18}; 1 \leq l_b \leq r_b \leq n$ ).

Запрос второго типа означает, что ваша программа готова дать ответ на задачу. После запроса второго типа необходимо вывести  $n$  целых чисел — загаданный массив.

Обратите внимание, ваша программа может сделать не более  $n$  запросов первого типа. При превышении данного ограничения, а также при попытке узнать сумму на *запрещенном* ранее отрезке, интерактор выведет «-1 -1 -1» и завершится с вердиктом WA. Чтобы не получить вердикт TL или PL, считав из ввода значения «-1 -1 -1», ваша программа должна завершить свою работу с нулевым кодом возврата.

### Пример

| стандартный ввод | стандартный вывод |
|------------------|-------------------|
| 3                |                   |
|                  | ? 1 1             |
| 1 2 2            |                   |
|                  | ? 3 3             |
| 3 2 3            |                   |
|                  | ? 1 3             |
| 6 1 2            |                   |
|                  | !                 |
|                  | 1 2 3             |

## Задача Н. Поиграем?

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 2 секунды         |
| Ограничение по памяти:  | 512 мегабайт      |

Это **интерактивная** задача.

1804 год. Вице-президент Соединённых Штатов Аарон Бёрр вызывает на дуэль кандидата в губернаторы штата Нью-Йорк Александра Гамильтона за серию оскорбительных памфлетов в свой адрес.

Но Бёрр разумный человек и понимает, что даже если он убьёт Гамильтона на дуэли, он потеряет свою репутацию, и его политическая карьера будет закончена. Поэтому противники решили просто сыграть в игру. Для честности они решили сыграть в неё  $g$  раз.

Каждую игру Гамильтон загадывает целое положительное число  $n$ , а Бёрр пытается его отгадать. Для любого целого положительного  $x$  Бёрр может спросить у Гамильтона долю чисел между 1 и  $n$  включительно, которые делятся на  $x$ . Иными словами, спрашивая про  $x$ , он получает значение выражения

$$\frac{\lfloor \frac{n}{x} \rfloor}{n},$$

причём Гамильтон сообщает ему результат в виде **несократимой** дроби (здесь  $\lfloor r \rfloor$  обозначает результат округления вниз вещественного числа  $r$ ).

Помогите Бёрру найти ответ за некоторое заранее определённое число запросов.

### Протокол взаимодействия

При запуске решения на вход подается целое число  $g$  — число игр между Гамильтоном и Бёрром ( $1 \leq g \leq 1000$ ).

Для каждого теста зафиксировано число  $q$  ( $6 \leq q \leq 60$ ) — максимальное количество запросов в одной игре. Гарантируется, что  $q$  запросов достаточно, чтобы решить задачу. Эти числа не сообщаются программе участника, но ограничения на эти числа в различных подзадачах приведены в таблице системы оценивания. Если программа участника делает более  $q$  запросов программе жюри, на этом тесте она получает в качестве результата тестирования «Wrong answer».

Чтобы сделать запрос, следует вывести строку «X  $t$ », где  $t$  — целое положительное число ( $1 \leq t \leq 10^{18}$ ), для которого требуется узнать значение выражения

$$\frac{\lfloor \frac{n}{t} \rfloor}{n}$$

В ответ на каждый запрос программа получает через пробел два числа  $a$  и  $b$  — числитель и знаменатель этой дроби **после сокращения** — или число  $-1$  в случае, если программа превысила ограничение по числу запросов.

Если программа определила загаданное число, то она должна вывести строку «N  $t$ », где  $t$  — предполагаемый ответ ( $1 \leq t \leq 10^{18}$ ). Если ответ был правильный, то в ответ программа получает строку «Correct», а если неправильный, то она получает строку «Wrong».

После этого программа переходит к следующей игре, если они остались, иначе она должна завершиться.

Обратите внимание, в случае считывания числа  $-1$  или строки «Wrong» вы **обязательно** должны сразу завершить вашу программу. В противном случае вердикт тестирующей системы может быть некорректным, в частности, вы можете получить вердикт Run-time error или Time limit exceeded!

Гарантируется, что в каждом тесте загаданные числа фиксированы в самом начале игры и не изменяются в зависимости от ваших запросов.

### Система оценки

Тесты к этой задаче состоят из девяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп.

| Группа | Баллы | Дополнительные ограничения |
|--------|-------|----------------------------|
| 0      | 0     | $q = 60$                   |
| 1      | 30    | $q = 60$                   |
| 2      | 10    | $q = 30$                   |
| 3      | 4     | $q = 20$                   |
| 4      | 4     | $q = 15$                   |
| 5      | 5     | $q = 10$                   |
| 6      | 5     | $q = 9$                    |
| 7      | 11    | $q = 8$                    |
| 8      | 10    | $q = 7$                    |
| 9      | 21    | $q = 6$                    |

## Пример

| стандартный ввод | стандартный вывод |
|------------------|-------------------|
| 2                | X 2               |
| 1 2              | X 3               |
| 3 10             | X 5               |
| 1 5              | X 4               |
| 1 5              | X 6               |
| 1 10             | X 10              |
| 1 10             | X 11              |
| 0 1              | N 10              |
| Correct          | X 1               |
| 1 1              | X 2               |
| 0 1              | N 1               |
| Correct          |                   |

## Замечание

В первом примере  $g = 2$ . Приведены примеры запросов, по которым игрок угадывает, что в первой игре загадано число 10, а во второй 1. Эти же числа загаданы в первом тесте в тестирующей системе.

В точности соблюдайте формат выходных данных. После каждого вывода обязательно выводите один перевод строки и сбрасывайте буфер вывода — для этого используйте `flush(output)` на языке Паскаль или Delphi, `fflush(stdout)` или `cout.flush()` в C/C++, `sys.stdout.flush()` на языке Python, `System.out.flush()` на языке Java.

## Задача I. Новогодний и прямоугольный

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 3 секунды         |
| Ограничение по памяти:  | 256 мегабайт      |

Это **интерактивная** задача.

На Новый год Дед Мороз подарил Глебу то, о чём он уже давно мечтал — клетчатый квадрат размером  $n \times n$ . Подарок этот не простой, а с сюрпризом — внутри квадрата Дед Мороз выбрал некоторый непустой прямоугольник, и в каждую клетку этого прямоугольника он положил по мандарину.

Теперь, чтобы получить желанный подарок, Глебу нужно сыграть с Дедом Морозом в очень интересную игру. Глеб должен отгадать, в каком именно прямоугольнике находятся все мандарины, подаренные Дедом Морозом. Будем считать, что строки и столбцы занумерованы числами от 1 до  $n$  снизу вверх и слева направо. Глеб может производить два типа запросов:

- ?  $x_1 y_1 x_2 y_2$  ( $1 \leq x_1 \leq x_2 \leq n$ ,  $1 \leq y_1 \leq y_2 \leq n$ ) — в ответ на этот запрос Дед Мороз говорит, сколько мандаринок находится в прямоугольнике, левым нижним углом которого является клетка  $(x_1, y_1)$ , а правым верхним — клетка  $(x_2, y_2)$ ;
- !  $x_1 y_1 x_2 y_2$  ( $1 \leq x_1 \leq x_2 \leq n$ ,  $1 \leq y_1 \leq y_2 \leq n$ ) — когда Глеб уверен, что он точно знает, где находятся мандарины, он должен сделать запрос такого вида, чтобы сообщить свой ответ. При этом  $(x_1, y_1)$  соответствует предполагаемому расположению левого нижнего угла, а  $(x_2, y_2)$  — правого верхнего.

### Формат входных данных

При запуске решения на вход вашей программе подается одно число  $n$  ( $1 \leq n \leq 2 \cdot 10^9$ ) — размер квадрата.

Затем на каждый запрос типа “?” вам будет выдаваться количество мандаринок, находящихся в указанном вами прямоугольнике.

### Формат выходных данных

Вы должны выводить корректные запросы в формате, описанном выше. Последним должен следовать единственный запрос вида “!”, после чего ваша программа должна немедленно завершиться. Ваша программа должна произвести не больше  $q$  (параметр зависит от номера группы) запросов типа “?”. Обратите внимание, что последний запрос, выводящий ответ, не входит в данные  $q$  запросов.

В точности соблюдайте формат выходных данных. После вывода каждой строки сбрасывайте буфер вывода — для этого используйте команды `flush(output)` на языке Паскаль или `Delphi`, `fflush(stdout)` или `cout.flush()` в `C/C++`, `sys.stdout.flush()` на языке `Python`, `System.out.flush()` на языке `Java`.

### Примеры

| стандартный ввод | стандартный вывод |
|------------------|-------------------|
| 4                | ? 1 1 4 4         |
| 6                | ? 1 3 4 4         |
| 6                | ? 2 3 4 4         |
| 4                | ! 1 3 3 4         |

### Замечание

Пример в условии иллюстрирует взаимодействие с проверяющей программой. Для прохождения первого теста не обязательно производить такие же запросы, как в примере.

Тесты к этой задаче состоят из шести групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп.

| Группа | Тесты         | Баллы | Дополнительные ограничения |               | Комментарий      |
|--------|---------------|-------|----------------------------|---------------|------------------|
|        |               |       | $n$                        | $q$           |                  |
| 0      | 1 – 1         | 0     | $n = 4$                    | $q = 1000$    | Тест из условия. |
| 1      | 2 – 12        | 10    | $n \leq 10$                | $q = 10\,000$ |                  |
| 2      | 13 – 23       | 20    | $n \leq 100$               | $q = 10\,000$ |                  |
| 3      | 24 – 34       | 20    | $n \leq 10\,000$           | $q = 20\,000$ |                  |
| 4      | 35 – 45       | 25    | $n \leq 2 \cdot 10^9$      | $q = 128$     |                  |
| 5      | 46 – $\infty$ | 25    | $n \leq 2 \cdot 10^9$      | $q = 64$      |                  |

## Задача J. Угадай выражение

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 2 секунды         |
| Ограничение по памяти:  | 1024 мегабайта    |

### Это интерактивная задача.

Жюри загадало некоторое арифметическое выражение, содержащее  $n + 1$  переменных  $a_0, a_1, \dots, a_n$ , которое вычисляется по формуле:

$$(\dots(((a_0 \text{ op}_1 a_1) \text{ op}_2 a_2) \text{ op}_3 a_3) \dots \text{ op}_n a_n) \bmod 10^9 + 7,$$

где  $\text{op}_1, \text{op}_2, \dots, \text{op}_n$  — это арифметические операции, каждая из которых является либо операцией «+» (сложение), либо операцией «×» (умножение), а  $a_0, a_1, \dots, a_n$  — переменные.

Например, если значения переменных равны  $(a_0, a_1, a_2) = (1, 1, 2)$ , а операции равны  $(\text{op}_1, \text{op}_2) = (+, \times)$ , то в результате вычисления выражения мы получим значение  $((1 + 1) \times 2) \bmod 10^9 + 7 = 4$ .

Вы можете несколько раз задать некоторые значения каждой из переменных и попросить жюри вычислить значение выражения. После этого вы должны угадать, чему равны  $\text{op}_1, \text{op}_2, \dots, \text{op}_n$ .

### Протокол взаимодействия

В начале взаимодействия с программой жюри вы должны считать целое число  $n$  ( $1 \leq n \leq 4000$ ) — количество операторов в формуле.

После этого вы можете вывести не более 275 запросов вида: «?  $a_0 a_1 \dots a_n$ » ( $0 \leq a_i < 10^9 + 7$ ). После этого вы должны считать одно число — значение выражения для данных переменных.

Когда вы определите, чему равны  $\text{op}_1, \text{op}_2, \dots, \text{op}_n$ , вы должны вывести запрос вида: «!  $s$ », где  $s$  — строка, состоящая из  $n$  символов + или × (маленькая латинская буква x), где  $i$ -й символ строки соответствует операции  $\text{op}_i$ .

Не забывайте выполнять операцию `flush` после каждого запроса (в том числе и после последнего).

В случае, если вы выведете больше 275 запросов на вычисление выражения, либо не будете соблюдать описанный протокол взаимодействия с программой жюри, вы можете получить любой вердикт.

### Примеры

| стандартный ввод | стандартный вывод         |
|------------------|---------------------------|
| 2                | ? 1 1 2                   |
| 4                | ? 1 1 3                   |
| 6                | ! +x                      |
| 10               | ? 1 1 1 1 1 1 1 1 1 1     |
| 5                | ? 0 4 2 4 2 4 2 4 2 4 2   |
| 6224             | ? 1 2 3 4 5 6 7 8 9 10 11 |
| 640750           | ! ++xxx+x+xx              |

## Задача К. Игра с бинарной строкой

|                         |                   |
|-------------------------|-------------------|
| Имя входного файла:     | стандартный ввод  |
| Имя выходного файла:    | стандартный вывод |
| Ограничение по времени: | 1 секунда         |
| Ограничение по памяти:  | 256 мегабайт      |

### Это интерактивная задача.

Алиса и Боб играют в игру. У них есть строка длины  $n$ , каждый символ строки — это либо 0, либо 1. Игроки ходят по очереди, начинает Алиса. На каждом ходу игрок должен изменить **ровно** один символ строки: 0 меняется на 1, а 1 — на 0. После хода игрока должна получиться строка, которая раньше в игре никогда не встречалась (в том числе до всех ходов). Если игрок не может сделать ход, то он проигрывает.

Вам нужно выбрать, за кого (Алису или Боба) вы хотите играть, проверяющая система будет играть за другого игрока. Вы должны выиграть игру за выбранного игрока.

### Протокол взаимодействия

В начале вы должны считать  $n$  ( $1 \leq n \leq 15$ ) — длину строки — и строку  $s$  длины  $n$  — начальное состояние строки. После этого выведите «Alice» (если вы хотите играть за Алису) или «Bob» (если хотите играть за Боба).

В каждый свой ход вы должны вывести одно число  $p$  ( $0 \leq p \leq n$ ).

- $p = 0$  символизирует, что вы сдаётесь.  $1 \leq p \leq n$  — позиция символа, который вы меняет своим ходом. Позиции в строке нумеруются слева направо от 1 до  $n$ .

В каждый ход соперника вы должны считать одно число  $p$  ( $-1 \leq p \leq n$ ).

- $p = 0$  символизирует, что соперник сдаётся. В этом случае вы должны завершить выполнение программы.
- $p = -1$  символизирует, что ваш последний ход привёл к строке, которая ранее встречалась, либо вы совершили недопустимый ход. В этом случае вы также должны завершить выполнение программы, иначе вы можете получить произвольный вердикт.  $1 \leq p \leq n$  — позиция символа, который соперник меняет своим ходом.

Обратите внимание, что если вы выбрали Алису, то вы делаете первый ход, а если вы выбрали Боба, то вы делаете второй ход.

Не забудьте после каждого хода выполнять операцию ‘flush’.

Для сброса буфера вывода (то есть для операции ‘flush’) сразу после вывода хода и перевода строки нужно сделать:

- `fflush(stdout)` в языке C++;
- `System.out.flush()` в Java;
- `stdout.flush()` в Python;
- `flush(output)` в Pascal;
- смотрите документацию для других языков.

В случае, если вы не будете выполнять операцию ‘flush’ после каждого хода, либо не будете соблюдать формат взаимодействия с программой жюри, вы можете получить любой вердикт.

### Пример

| стандартный ввод | стандартный вывод |
|------------------|-------------------|
| 2                | Alice             |
| 00               | 2                 |
| 1                | 2                 |
| 0                |                   |