Twin Cookies

Название задачи	Twin Cookies
Входной файл	стандартный ввод
Выходной файл	стандартный вывод
Ограничение по времени	1 секунда
Ограничение по памяти	256 мегабайт

Это интерактивная задача. Ваша программа должна взаимодействовать с проверяющим модулем поочередно отправляя сообщения в стандартный вывод и читая сообщения из стандартного ввода.

Софи готовится праздновать день рождения своих близнецов. Близнецы любят печеньки. Для празднования дня рождения они хотят попробовать что-то новенькое: печеньки из Unique Cookie Tastiness Company (UCTC, Компании Уникальных Вкусов Печенек).

Каждая печенька, произведенная UCTC, имеет целочисленное вкусовое значение в диапазоне от 1 до 10^{16} включительно.Так как близнецы Софи ревнуют друг к другу, каждый из них должен получить печеньки с одинаковой суммой вкусовых значений.

UCTC принимает заказы только на **ровно** n печенек. В каждом заказе клиент указывает желаемое вкусовое значение каждой из n печенек.

В соответствии со своим названием, Unique Cookie Tastiness Company отказывается производить две печеньки с одинаковым вкусовым значением одному и тому же клиенту. Софи должны быть уверена, что она никогда не заказвала одно и то же вкусовое значение дважды -- ни в одном заказе, ни в различных. Софи никогда не покупала у UCTC ранее, так что она может заказать каждое возможное вкусовое значение один раз.

Есть ещё одно препятствие на пути Софи: Хорошо известно, что служба доставки UCTC ужасна. Когда клиент заказывает n печенек, только одна из этих n печенек действительно доходит до клиента. Остальные съедаются по дороге работниками службы доставки. Клиент не может повлиять на то, какая из n заказанных печенек будет ему доставлена.

Так как день рождения быстро приближается, Софи может сделать не более 101 заказа. Ваша задача помочь ей.

Конкретнее, Вы должны сделать следующее:

1. Во-первых, закажите печеньки. Вы можете сделать не более 101 заказа, каждый из которых состоит из ровно n желаемых вкусовых значений. Вы делаете один заказ за один раз. **Немедленно после каждого заказа Вы получаете вкусовое значение доставленной печеньки из этого заказа.**

   Помните, что Вам нельзя использовать одно и тоже вкусовое значение дважды, даже если это разные заказы. (В частности, если Вы заказали печеньку с вкусовым значением t но она не была доставлена, Вы **не можете** заказать печеньку с таким вкусовым значением снова.)

2. Далее, разделите печеньки. Как только Вы получите достаточно печенек, Вы должны разделить**некоторые** из доставленных печенек между близнецами. Каждый из близнецов должен получить как минимум одну печеньку, и каждый из близнецов должен получить печеньки с одинаковой суммой значений вкусов. **Вы не обязаны использовать все доставленные печеньки!**

Output (выходные данные)

Каждый раз, когда Ваша программа выводит одну или несколько строк в стандартный вывод, Вы должны завершить действие **очисткой потока вывода**. Это необходимо, чтобы быть уверенным, что ваши данные будут доставлены проверяющему модулю немедленно.

Примеры, как это должно быть сделано:

- В C++ есть несколько опций.
  - `fflush(stdout);`
  - `std::cout << std::flush;`

    ```
      - `std::cout << std::endl;` (обратите внимание, что это такж
    е выведет еще одну новую строку)
    ```

  - чтение из `std::cin` также очищает поток вывода
- на Java, Вы можете использовать `System.out.flush()`
- на Python, Вы можете использовать `sys.stdout.flush()`

Interaction Protocol (протокол взаимодействия)

Ваша программа должнв выполнить следующую последовательность действий:

1. Прочитать значение n из стандартного ввода.
2. Не более 101 раз:
    1. Во-первых, вывести одну строку, описывающую заказ на n печенек в стандартный вывод.
    2. Затем, прочитать вкусовое значение доставленной печенькииз стандартного ввода. Гарантируется, что это значение было среди n вкусовых значений этого заказа.
3. Выведите три строки, описывающие один из способов разделить доставленные печеньки между близнецами.

Проверяющий модуль будет выводить каждое целое число в новой строке.

Чтобы заказать печеньки, выведите одну строку с ? и последующими за ним n целыми числами: вкусовые значения печенек, которые Вы хотите заказать. Выводите один пробел перед каждым их этих n целых чисел.

Помните, что Вы можете сделать не более 101 заказов и Вам не разрешено использовать одно и то же вкусовое значение дважды.

Как только Вы заказали достаточно печенек, выведите три строки, описывающие распределение печенек между близнецами Софи.

Первая из этих линий должна содержать "! m k", где $m, k > 0$: число печенек которые должен получить первый и второй близнецы, соответственно.

Вторая из этих строк должна содержать m целых чисел, разделенных пробелом: вкусовые значения печенек первого близнеца.

Аналогично, третья строка должна содержать k целых чисел, разделенных пробелом: вкусовые значения печенек второго близнеца.

Ответ должен удовлетворять следующим условиям:

1. Каждый близнец должен получить как минимум одну печеньку.
2. Каздый близнец должен получить печеньки с одинаковой суммой вкусовых значений.
3. Могут быть использованы только доставленные печеньки.
4. Каждая из этих печенек может достаться только одному из близнецов.

Принимается любой ответ, удовлетворяющий этим условиям. В частности, Вы можете выводить выбранные печеньки в любом порядке.

После вывода финальных трёх строк, очистите поток выводапоследний раз и затем **нормально завершите программу**.

Scoring (оценка)

Подзадача 1 (8 баллов): $n = 1$

Подзадача 2 (9 баллов): $1 \leq n \leq 2$

Подзадача 3 (18 баллов): $1 \leq n \leq 25$

Подзадача 4 (16 баллов): $1 \leq n \leq 200$

Подзадача 5 (13 баллов): $1 \leq n \leq 1000$

Подзадача 6 (36 баллов): $1 \leq n \leq 5000$

Examples (пример)

standard input	standard output
1 13 7 31 12 5 3	? 13 ? 7 ? 31 ? 12 ? 5 ? 3 ! 2 3 7 13 12 5 3
2 7 2 5	? 3 7 ? 2 8 ? 1 5 ! 2 1 2 5 7

Note (пояснение)

Примеры ввода и вывода нужно читать построчно. Ваша программа поочередно читает одно значение из стандартного ввода и пишет одну строку (или три строки в конце) в стандартный вывод.

Проверяющий модуль выбирает доставляемую печеньку произвольно. Это значит, что проверяющий модуль может подстраиваться под Ваши заказы в некоторых тестах, но в некоторых тестах значение может быть и случайным. В частности, для $n = 2$, если Вы сделаете такой же набор заказаов как во втором примере,Вы можете получить другой набор печенек.