

## Задача А. Целочисленное деление

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

Заданы два целых числа  $A$  и  $B$ . Результатом целочисленного деления  $A$  на  $B$  будем считать целое число  $Q$  такое, что  $A = Q \cdot B + R$  и при этом  $0 \leq R < |B|$ . Выведите  $Q$ .

### Формат входных данных

В первой строке ввода заданы два целых числа  $A$  и  $B$  ( $-2^{31} \leq A, B < 2^{31}$ ).

### Формат выходных данных

Выведите одно число — результат целочисленного деления  $A$  на  $B$ . Если  $B = 0$ , выведите вместо этого строку «Na nol' delit' nel'zya!!!».

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
2 3	0
239 0	Na nol' delit' nel'zya!!!

## Задача В. Лишнее число

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

В штаб секретной службы поступило сообщение от одного из агентов. Поступившее сообщение в зашифрованном виде представляет собой последовательность чисел, и лишь специальная программа способна расшифровать его и получить связный текст.

Обычно программа-расшифровщик быстро и бесшумно выдаёт связистам расшифрованный текст, но в этот раз вместо текста от программы поступил сигнал тревоги, свидетельствующий о том, что при пересылке сообщение было взломано или просто повреждено.

Корректное зашифрованное сообщение — это последовательность из  $4 \cdot k$  целых чисел, в котором  $k$  различных чисел присутствуют по 4 раза каждое; для расшифровки даже не важны значения этих чисел, а важен лишь их порядок.

Однако, изучив зашифрованное сообщение, связисты обнаружили, что в нём  $4 \cdot k + 1$  число. При этом ровно одно число является «лишним», то есть при его удалении зашифрованное сообщение становится корректным сообщением из  $4 \cdot k$  чисел (возможно, четыре из них равны удалённому числу).

Связисты решили, что на будущее им нужна программа, которая находит такое «лишнее» число автоматически. Помогите им написать такую программу.

### Формат входных данных

В первой строке задано число  $n = 4 \cdot k + 1$ , где  $n$  и  $k$  целые, и  $1 \leq k \leq 10\,000$ . Во второй строке находятся числа  $a_1, a_2, \dots, a_n$ , разделённые пробелами — зашифрованное сообщение. Известно, что  $0 \leq a_i \leq 1\,000\,000$ .

### Формат выходных данных

В первую строку выведите «лишнее» число из набора  $A_i$ .

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
5 4 1 4 4 4	1
9 1 3 3 1 3 3 3 1 1	3

## Задача С. Разность между значениями

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

В выражении

$$f(x) = ax + \frac{x}{b}$$

известны коэффициенты  $a$  и  $b$ . Найдите разность значений функции  $f(x)$  в двух точках  $x_1$  и  $x_2$  при условии, что  $|x_2 - x_1| = 1$ .

### Формат входных данных

В первой строке заданы через пробел четыре целых числа  $a, b, x_1$  и  $x_2$  ( $1 \leq a, b, x_1, x_2 \leq 10^9$ ). Гарантируется, что числа  $x_1$  и  $x_2$  отличаются ровно на единицу.

### Формат выходных данных

Выведите одно число — требуемую разность  $f(x_1) - f(x_2)$ . Абсолютная погрешность должна составлять не более  $10^{-5}$ .

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
1 2 3 4	-1.5
2 1 4 3	3

## Задача D. Сумма значений функции

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

Найдите сумму значений функции

$$f(x) = x + \frac{1}{x}$$

в нескольких целых точках.

### Формат входных данных

В первой строке задано целое число  $n$  — количество точек ( $1 \leq n \leq 50$ ). В следующей строке заданы  $n$  целых чисел  $x_1, x_2, \dots, x_n$  через пробел — точки, значения функции в которых нужно просуммировать ( $0 < |x_i| \leq 10^9$ ).

### Формат выходных данных

Выведите одно число — сумму значений функции  $f(x)$  в заданных точках. Ответ считается правильным, если абсолютная или относительная погрешность не превышает  $10^{-9}$ .

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3 1 2 3	7.833333333333333
2 1 -1	0

## Задача Е. Представление 16-битного целого числа

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

Задано целое число. Выведите представление этого числа в 16-битном двоичном дополнительном коде.

### Формат входных данных

В первой строке задано одно целое число  $n$  в десятичной записи ( $-2^{15} \leq n < 2^{15}$ ).

### Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись дополнительного кода для числа  $n$ . Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3	00000000 00000011 00 03
-57	11111111 11000111 FF C7

## Задача F. Представление 32-битного целого числа

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

Задано целое число. Выведите представление этого числа в 32-битном двоичном дополнительном коде.

### Формат входных данных

В первой строке задано одно целое число  $n$  в десятичной записи ( $-2^{31} \leq n < 2^{31}$ ).

### Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись дополнительного кода для числа  $n$ . Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3	00000000 00000000 00000000 00000011 00 00 00 03
-57	11111111 11111111 11111111 11000111 FF FF FF C7

## Задача G. Представление 32-битного вещественного числа

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

Задано вещественное число. Выведите представление этого числа в 32-битном типе `float`. В этом типе 1 старший бит хранит знак числа, следующие 8 битов отведены под экспоненту, а в оставшихся 23 битах находится мантисса.

### Формат входных данных

В первой строке заданы два целых числа  $p$  и  $q$  через пробел ( $|p|, |q| \leq 10^4, q \neq 0$ ). Число, которое нужно представить в типе `float` — это  $\frac{p}{q}$ . Помните, что если результат деления нельзя сохранить точно, в результирующей переменной типа `float` должно оказаться наиболее близкое из тех чисел, которые в этом формате можно хранить.

### Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись представления данного числа. Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

Число 0 следует записывать с положительным (то есть нулевым) знаковым битом.

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
9 2	01000000 10010000 00000000 00000000 40 90 00 00
5 -3	10111111 11010101 01010101 01010101 BF D5 55 55

## Задача Н. Представление 64-битного вещественного числа

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

Задано вещественное число. Выведите представление этого числа в 64-битном типе `double`. В этом типе 1 старший бит хранит знак числа, следующие 11 битов отведены под экспоненту, а в оставшихся 52 битах находится мантисса.

### Формат входных данных

В первой строке заданы два целых числа  $p$  и  $q$  через пробел ( $|p|, |q| \leq 10^9, q \neq 0$ ). Число, которое нужно представить в типе `double` — это  $\frac{p}{q}$ . Помните, что если результат деления нельзя сохранить точно, в результирующей переменной типа `double` должно оказаться наиболее близкое из тех чисел, которые в этом формате можно хранить.

### Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись представления данного числа. Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

Число 0 следует записывать с положительным (то есть нулевым) знаковым битом.

### Примеры

<i>стандартный ввод</i>
9 2
<i>стандартный вывод</i>
01000000 00010010 00000000 00000000 00000000 00000000 00000000 00000000 40 12 00 00 00 00 00 00
<i>стандартный ввод</i>
5 -3
<i>стандартный вывод</i>
10111111 11111010 10101010 10101010 10101010 10101010 10101010 10101011 BF FA AA AA AA AA AA AB

## Задача I. Квадратура круга

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

— Извини, Теодор, но это ты очень странно рассуждаешь. Бессмыслица — искать решение, если оно и так есть. Речь идёт о том, как поступать с задачей, которая решения не имеет. Это глубоко принципиальный вопрос, который, как я вижу, тебе, прикладнику, к сожалению, не доступен.

Аркадий и Борис Стругацкие, «Понедельник начинается в субботу»

*Квадратура круга* — задача, заключающаяся в нахождении построения с помощью циркуля и линейки квадрата, равновеликого данному кругу (то есть имеющего ту же площадь, что и круг). Наряду с трисекцией угла и удвоением куба, эта задача является одной из самых известных неразрешимых задач на построение с помощью циркуля и линейки. Однако, задача о квадратуре круга становится разрешимой, если расширить средства построения, а также если искать не точное, а приближённое решение.

В этой задаче требуется по кругу, заданному координатами центра и радиусом, построить квадрат, площадь которого отличается от площади этого круга не более чем на  $10^{-6}$ . В качестве средства предлагается использовать компьютер и один из доступных языков программирования.

Напомним, что площадь квадрата со стороной  $a$  равна  $a^2$ , а площадь круга радиуса  $r$  равна  $\pi \cdot r^2$ , где  $\pi \approx 3.1415926535897932384626433832795\dots$  — это половина длины окружности единичного радиуса.

### Формат входных данных

В единственной строке заданы через пробел три целых числа  $x$ ,  $y$  и  $r$  ( $|x|, |y| \leq 100$ ,  $1 \leq r \leq 100$ ) — координаты центра круга и его радиус.

### Формат выходных данных

Выведите четыре строки. Каждая строка должна содержать два числа через пробел — координаты одной из вершин квадрата. Найденный квадрат должен иметь стороны, параллельные осям координат, и площадь, равную площади данного круга, а его центр должен совпадать с центром круга. В первой строке выведите координаты левой нижней вершины квадрата, во второй — левой верхней, в третьей — правой верхней и в четвёртой — правой нижней вершины.

Выводите вещественные числа как можно более точно! Допускается экспоненциальная форма вывода. При проверке ответов **все** проверки на равенство — сравнение координат точек и площадей квадратов — производятся с точностью до  $10^{-6}$ .

### Пример

<i>стандартный ввод</i>	
2	3 5
<i>стандартный вывод</i>	
-2.431134627264	-1.431134627264
-2.431134627264	7.431134627264
6.431134627264	7.431134627264
6.431134627264	-1.431134627264

## Задача J. Стандартное нормальное распределение

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

В теории вероятностей *стандартным нормальным распределением* называется распределение, плотность которого равна

$$\phi(x) = \frac{1}{\sqrt{2\pi}} \exp -\frac{x^2}{2},$$

а функция распределения имеет вид

$$\Phi(x) = \int_{-\infty}^x \phi(t) dt.$$

Нормальное распределение играет важную роль в теории вероятностей. Как и любая функция распределения,  $\Phi(x)$  стремится к нулю при  $x \rightarrow -\infty$  и к единице при  $x \rightarrow +\infty$ .

Интересно отметить, что выражение для  $\Phi(x)$  содержит интеграл, который «не берётся», то есть не представляется в виде выражения, использующего лишь элементарные функции. Несмотря на это, для конкретных значений  $x$  можно вычислять значения  $\Phi(x)$  с любой точностью.

По данному аргументу  $x$  найдите приближённое значение  $\Phi(x)$ .

### Формат входных данных

В первой строке заданы через пробел два целых числа  $p$  и  $q$  ( $-10\,000 \leq p \leq 10\,000$ ,  $1 \leq q \leq 10\,000$ ). Число  $x$  получается как отношение  $\frac{p}{q}$ .

### Формат выходных данных

Выведите одно число — значение  $\Phi(x)$ . Абсолютная погрешность должна составлять не более  $10^{-7}$ .

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
1 2	0.6914624432
2 1	0.977249857

## Задача К. Сложение и переворачивание

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

Рассмотрим целое неотрицательное число  $x$ , которое хранится в 32 битах памяти:

$$x = b_{31} \cdot 2^{31} + b_{30} \cdot 2^{30} + \dots + b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0,$$

где каждый бит  $b_i$  может независимо от остальных принимать два значения: 0 и 1.

Произведём над числом последовательность операций, возможно, пустую. За одну операцию разрешается либо прибавить к числу единицу, либо развернуть составляющие его биты: 31-й бит поменять местами с нулевым, 30-й с первым, ..., 16-й с 15-м. Можно выполнить любое количество любых операций в любом порядке.

За какое минимальное количество операций можно из нуля получить заданное число  $n$ ?

Сложение производится по модулю  $2^{32}$ , то есть, если текущее число равно  $2^{32} - 1$ , то после прибавления единицы число станет нулём.

### Формат входных данных

В единственной строке задано целое число  $n$  ( $0 \leq n < 2^{32}$ ).

### Формат выходных данных

Выведите одно число: минимальное количество операций, которое потребуется, чтобы получить из нуля заданное число  $n$ .

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
5	5
2147483648	2

### Пояснения к примерам

В первом примере самый быстрый способ получить число 5 — это пять раз прибавить единицу.

Во втором примере сначала получим из нуля единицу, а затем развернём составляющие число биты, превратив  $1 = 2^0$  в  $2\,147\,483\,648 = 2^{31}$ .

## Задача L. Перестановка битов

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

В этой задаче требуется быстро переставлять биты в машинном представлении чисел.

Знаете ли вы, как представляются числа в современных компьютерах? Один из удобных способов хранить не очень большое по модулю целое число — представить его в типе `int32`. В этом типе каждому числу предоставляется в распоряжение 32 бита, нумерующихся целыми числами от 0 до 31. Каждый бит может быть либо нулём, либо единицей. Если значения битов равны  $b_0, b_1, b_2, \dots, b_{30}, b_{31}$ , то само число получается как сумма

$$b_0 \cdot 2^0 + b_1 \cdot 2^1 + b_2 \cdot 2^2 + \dots + b_{30} \cdot 2^{30} - b_{31} \cdot 2^{31}.$$

Обратите внимание на то, что последнее слагаемое имеет отрицательный знак. В этом типе можно представить любое целое число от  $-2^{31}$  до  $2^{31} - 1$ .

У числа, представленного в типе `int32`, можно переставить биты. Рассмотрим перестановку  $p_0, p_1, p_2, \dots, p_{30}, p_{31}$ , состоящую из целых чисел от 0 до 31, каждое из которых встречается в ней ровно один раз. После перестановки битов  $x_0, \dots, x_{31}$  числа  $x$  в соответствии с  $p$  получается число  $y = p(x)$ , состоящее из битов  $y_{p_0} = x_0, \dots, y_{p_{31}} = x_{31}$ .

Знаете ли вы, откуда берутся «случайные» числа? Один из простых способов получения псевдослучайных чисел — линейный конгруэнтный генератор. Такой генератор характеризуется константами  $a$  (множитель),  $c$  (добавка) и  $m$  (модуль), а также состоянием  $s$ . При генерации следующего числа сначала производится операция  $s \leftarrow (s \cdot a + c) \bmod m$ , а затем новое значение состояния  $s$  объявляется следующим псевдослучайным числом. Конечно, у такого генератора есть период, не превосходящий  $m$ : как только в  $s$  получилось число, которое уже встречалось ранее, все дальнейшие действия будут давать те же результаты, что и раньше.

Для ускорения работы такого генератора можно избавиться от операции взятия остатка по модулю  $m$ . Числа, получающиеся после операции  $s \leftarrow (s \cdot a + c)$ , будем представлять в типе `int32`. При этом некоторая часть числа может теряться, но у того, что останется, будет такой же остаток от деления на  $2^{32}$ , что и у настоящего результата. Фактически при таком использовании можно считать, что  $m = 2^{32}$ , но из верхней половины возможных остатков мы вычитаем  $2^{32}$ .

Заданы числа  $n, a, c$  и  $s$ , а также перестановка битов  $p$ . Используя линейный конгруэнтный генератор в типе `int32` с параметрами  $a$  и  $c$  и начальным состоянием  $s$ , сгенерируйте следующие  $n$  псевдослучайных чисел  $x_1, x_2, \dots, x_n$ . К каждому из этих чисел примените перестановку битов  $p$ , после чего сложите все полученные числа. Будьте внимательны: несмотря на то, что каждое из полученных чисел представимо в типе `int32`, их сумма может не быть в нём представима, поэтому для суммирования следует использовать более широкий тип данных.

### Формат входных данных

В первой строке ввода заданы четыре целых числа  $n, a, c$  и  $s$  — количество операций и параметры линейного конгруэнтного генератора ( $1 \leq n \leq 100\,000\,000$ , а числа  $a, c$  и  $s$  могут быть любыми представимыми в типе `int32`). Гарантируется, что период генератора равен  $2^{32}$ . Во второй строке задана перестановка битов  $p$  — числа  $p_0, p_1, \dots, p_{31}$ , среди которых каждое целое число от 0 до 31 встречается ровно один раз. Соседние числа в строках разделяются пробелом.

### Формат выходных данных

Выведите одно целое число: сумму  $n$  полученных псевдослучайных чисел, к каждому из которых применена перестановка битов  $p$ .



## Задача М. Части плоскости

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

Даны  $N$  точек на плоскости. Проведём прямые через каждую пару точек. На сколько частей эти прямые делят плоскость?

### Формат входных данных

В первой строке задано число  $N$  — количество точек ( $2 \leq N \leq 10$ ). Следующие  $N$  строк содержат по два числа  $X_i$  и  $Y_i$  каждая через пробел — координаты  $i$ -й точки ( $-100 \leq X_i, Y_i \leq 100$ ). Никакие две данные точки не совпадают, никакие три не лежат на одной прямой. Все числа во входных данных целые.

### Формат выходных данных

В первой строке выведите  $P$  — количество частей, на которые полученные прямые делят плоскость.

### Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
4 0 0 0 1 1 0 1 1	16
3 1 5 2 3 -8 4	7

## Задача N. Преобразования плоскости

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

Рассмотрим последовательность преобразований плоскости, каждое из которых состоит в том, чтобы повернуть и масштабировать плоскость относительно фиксированной точки так, чтобы один фиксированный вектор, направленный из этой точки, переходил в другой вектор.

Даны точка на плоскости и последовательность преобразований указанного вида. В какую точку плоскости перейдёт данная точка после последовательного применения всех преобразований?

### Формат входных данных

В первой строке записаны через пробел три целых числа  $n$ ,  $x$  и  $y$  ( $1 \leq n \leq 100$ ,  $|x|, |y| \leq 100$ ). Следующие  $n$  строк содержат описания преобразований в порядке их применения; в  $i$ -й из этих строк записаны через пробел шесть целых чисел  $p_x^{(i)}$ ,  $p_y^{(i)}$ ,  $u_x^{(i)}$ ,  $u_y^{(i)}$ ,  $v_x^{(i)}$  и  $v_y^{(i)}$ , означающих, что плоскость поворачивается и масштабируется относительно точки  $(p_x^{(i)}, p_y^{(i)})$  так, что вектор  $u^{(i)} = (u_x^{(i)}, u_y^{(i)})$  переходит в вектор  $v^{(i)} = (v_x^{(i)}, v_y^{(i)})$  (другими словами, точка  $(p_x^{(i)} + u_x^{(i)}, p_y^{(i)} + u_y^{(i)})$  переходит в точку  $(p_x^{(i)} + v_x^{(i)}, p_y^{(i)} + v_y^{(i)})$ ) ( $|p_x^{(i)}|, |p_y^{(i)}|, |u_x^{(i)}|, |u_y^{(i)}|, |v_x^{(i)}|, |v_y^{(i)}| \leq 100$ , векторы  $u^{(i)}$  и  $v^{(i)}$  — ненулевые).

### Формат выходных данных

В первой строке выведите два числа через пробел — координаты точки, в которую переместится точка  $(x, y)$  в результате данных преобразований. Координаты должны иметь относительную или абсолютную погрешность не более  $10^{-9}$ . Допускается экспоненциальная форма вывода ответа.

### Примеры

<b>стандартный ввод</b>
1 2 0 0 0 1 0 0 2
<b>стандартный вывод</b>
0 4
<b>стандартный ввод</b>
2 0 0 3 5 -1 1 1 -1 5 3 1 -1 -1 1
<b>стандартный вывод</b>
4.0000 -4.0000
<b>стандартный ввод</b>
1 1 3 0 0 5 1 1 -1
<b>стандартный вывод</b>
8.461538461538e-001 2.307692307692e-001