Задача 1. Параметры командной строки

Источник:	базовая I
Имя входного файла:	`argv`
Имя выходного файла:	`stdout/stderr`
Ограничение по времени:	разумное
Ограничение по памяти:	разумное

Внимание: эта задача проверяется на **emailtester**, а не в nsuts.
Код задачи: `1/1_cmdargs`

В этой задаче нужно выполнить одну арифметическую операцию: сложение, вычитание или умножение. В некоторых случаях нужно делать эту операцию по модулю, в некоторых — нет.

Входные данные передаются через параметры командной строки. Операция определяется типом и двумя аргументами, которые идут в командной строке подряд в точности в этом порядке.

Тип всегда является одним из:

- `add` — нужно сложить два числа,
- `sub` — нужно вычесть из первого числа второе,
- `mul` — нужно найти произведение двух чисел.

Например, если исполняемый файл вашего решения называется `sol.exe`, то следующая команда предписывает вычесть из числа 173 число 546 (результат равен -373):

```
sol.exe sub 173 546
```

Кроме того, в командной строке может быть задан модуль, по которому нужно выполнять операцию. Модуль задаётся параметром `-m` и числом, которое задаёт модуль — они также всегда идут подряд. Модуль может быть задан либо в самом начале командной строки, либо в самом конце. Например:

```
sol.exe sub 135 1793846 -m 100

sol.exe -m 100 sub 135 1793846
```

Гарантируется, что других параметров в командной строке нет. Все числа целые, не превышают 10^9 по абсолютной величине. Число, задающее модуль, также положительно и больше единицы.

Одно целое число, результат арифметической операции, нужно вывести в стандартный поток вывода `stdout`. Если операция выполняется по по модулю, то это число должен быть неотрицательным и меньше модуля. Кроме того, решение должно завершиться с нулевым кодом возврата.

Пользователи часто запускают программы без параметров, и от вас требуется также поддержать этот случай. Если решение запускается без параметров, то оно должно:

- ничего не выводить в `stdout`,
- вывести сообщение `No parameters specified.` в стандартный поток ошибок `stderr`,
- вернуть код возврата 13, сообщая таким образом о неверном использовании.

Примеры

argv	stdout/stderr
`sol.exe mul 1000000000 17345`	`17345000000000`
`sol.exe`	`No parameters specified.`
`sol.exe sub 135 1793846 -m 100`	`89`

Задача 2. Обращение по модулю

Источник:	базовая I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Число B называется обратным к A по модулю M, если:

1. $A \cdot B$ имеет остаток 1 при делении на M, т.е. $(A \cdot B - 1)$ делится на M.
2. $0 \leqslant B < M$

Формат входных данных

В первой строке входного файла заданы через пробел два целых числа T ($1 \leqslant T \leqslant 10^5$) и M ($2 \leqslant M < 10^9$) — количество тестов и модуль M соответственно. Гарантируется, что число M — простое.

Следующие T строк содержат по одному целому числу A ($0 \leqslant A < M$)

Формат выходных данных

Для каждого входного числа выведите в отдельной строке обратный элемент к нему, либо -1 если обратного не существует.

Пример

input.txt	output.txt
4 13 8 7 1 0	5 2 1 -1

Задача 3. Параметры командной строки 2

Источник:	базовая II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	разумное
Ограничение по памяти:	разумное

Код задачи: `1/3_easyargs`

В этой задаче предлагается реализовать алгоритм выделения параметров из командной строки в простом случае.

Командная строка будет содержать только следующие символы:

- латинские буквы (большие и маленькие) и цифры,
- пробел (ASCII 32),
- двойные кавычки (ASCII 34).

Параметры записываются в командную строку подряд в порядке следования. Между каждыми двумя соседними параметрами вставляется один или несколько пробелов, чтобы отделить их друг от друга. Параметр может быть целиком заключён в двойные кавычки. Если внутри параметра есть пробелы, он заключён в двойные кавычки **обязательно**. Символов двойных кавычек в самих параметрах нет — ими лишь можно окружать параметры. Все параметры непустые.

Командная строка, которую необходимо разобрать, записана в единственной строке входного файла. Её длина не превышает 100 символов. Учтите, что в этой командной строке отсутствует имя исполняемого файла. В выходной файл нужно вывести искомые параметры командной строки, по одному параметру в строке. Каждый параметр должен быть заключён в квадратные скобки. Особый нулевой параметр (саму командную строку) выводить не нужно.

Данная задача проверяется на **emailtester**. Более того, она проверяется на четырёх разных компиляторах:

1. `cl` — компилятор C из Microsoft Visual Studio (основной компилятор в системе Windows).
2. `gcc` — компилятор C из GNU Compiler Collection (основной компилятор в системе Linux), сборка Mingw-w64.
3. `clang` — компилятор C из Clang для LLVM (основной компилятор в системах MacOSX и FreeBSD), команда `clang.exe` из официальной виндовой сборки.
4. `tcc` — малоизвестный и очень забавный компилятор Tiny C Compiler.

Решение должно компилироваться и верно работать на всех четырёх компиляторах.

Примеры

input.txt	output.txt
`"abc d" e`	`[abc d]` `[e]`

`"long p "  "p 2" "sht"  s pars`	`[long p ]`
	`[p 2]`
	`[sht]`
	`[s]`
	`[pars]`

Задача 4. Биномиальные коэффициенты

Источник:	основная I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	5 секунд
Ограничение по памяти:	разумное

Внимание: эта задача проверяется на **emailtester**, а не в nsuts.
Код задачи: `1/4_bincomb`

Отправлять нужно **весь проект** Visual C++ (версии 2013 или новее), то есть:

- файл проекта (имеет расширение `.vcxproj`),
- все файлы с исходным кодом (имеют расширения `.c` и `.h`).

Поскольку файлов может быть много, удобнее запаковать эти файлы в `.zip` архив и приложить один архив.

Можно также включать другие файлы в отправку. Однако имейте ввиду, что Visual Studio порождает много шлака, который не нужен, и проверяющая система будет на него ругаться. Примерный список того, что отправлять точно не нужно, можно найти здесь.

При создании проекта Visual Studio прописывает настройки по умолчанию. Некоторые из них менять нельзя:

1. Конфигурация `Release` должна присутствовать — она проверяется.
2. Целевая платформа `Win32` должна присутствовать — она проверяется.
3. Выходные файлы после сборки должны попадать в поддиректорию `Release`.

Рекомендуется самостоятельно проверить архив перед отправкой: распакуйте его в новое место и проверьте, что в нём всё нормально собирается и запускается.

Формат входных данных

В первой строке входного файла задано два целых числа: M — модуль и T — количество запросов ($2 \leqslant M \leqslant 10^9$, $1 \leqslant T \leqslant 10^6$).

В каждой из следующих T строк задано два целых числа N и K, для которых нужно вычислить биномиальный коэффициент ($0 \leqslant N \leqslant 2\,000$, $|K| \leqslant 10^9$).

Формат выходных данных

Для каждого из T запросов нужно вывести ответ в отдельной строке. Ответ — это биномиальный коэффициент из N по K по модулю M (целое неотрицательное число).

Примеры

input.txt	output.txt
17 12 0 0 10 5 3 -1 5 9 6 6 13 2 5 0 5 1 5 2 5 3 5 4 5 5	1 14 0 0 1 10 1 5 10 10 5 1

Задача 5. Наибольший общий делитель

Источник:	основная I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Наибольшим общим делителем целых чисел A и B называется такое целое число D, что:

1. $D > 0$;
2. A и B делятся на D без остатка;
3. D максимально.

Формат входных данных

В первой строке входного файла задано целое число T ($1 \leqslant T \leqslant 10^5$) — количество тестов. Следующие T строк содержат по два целых числа A и B ($1 \leqslant A, B \leqslant 10^9$).

Формат выходных данных

Для каждой входной пары чисел выведите в отдельной строке их наибольший общий делитель.

Пример

input.txt	output.txt
4 1 1 12 12 2 3 12 18	1 12 1 6

Задача 6. Биномиальный коэффициент 2

Источник:	основная I
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	10 секунд
Ограничение по памяти:	разумное

Требуется вычислить биномиальный коэффициент из N по K.

Формат входных данных

В первой строке входного файла задано целое число T ($1 \leqslant T \leqslant 10^5$) — количество тестов.

В каждой из следующих T строк задано два целых числа N и K, для которых нужно вычислить биномиальный коэффициент ($0 \leqslant K \leqslant N \leqslant 10^6$).

Формат выходных данных

Для каждого из T запросов нужно вывести ответ в отдельной строке. Ответ — это биномиальный коэффициент из N по K по модулю $1\,000\,000\,007$ ($10^9 + 7$).

Пример

input.txt	output.txt
2 5 2 100 50	10 538992043

Задача 7. Система линейных уравнений

Источник:	основная II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	разумное
Ограничение по памяти:	разумное

Дана система из N линейных уравнений и N переменных:

$$a_{1,1} \cdot x_1 + a_{1,2} \cdot x_2 + \ldots + a_{1,N} \cdot x_N = b_1$$
$$a_{2,1} \cdot x_1 + a_{2,2} \cdot x_2 + \ldots + a_{2,N} \cdot x_N = b_2$$
$$\vdots$$
$$a_{N,1} \cdot x_1 + a_{N,2} \cdot x_2 + \ldots + a_{N,N} \cdot x_N = b_N$$

Требуется решить данную систему методом Гаусса. Гарантируется, что данная система имеет единственное решение.

Формат входных данных

В первой строке задано число N — количество уравнений и переменных ($1 \leqslant N \leqslant 200$).

Следующие N строк содержат описывают систему уравенений. i-ая строка содержит по (N+1) вещественному числу: коэффициенты $a_{i,N}$, $a_{i,N}$, …, $a_{i,N}$ и значение правой части системы b_i.

Формат выходных данных

Выведите N строк: значения переменных x_1, x_2, …, x_N.

Ответ будет считаться правильным, если абсолютная погрешность в каждой строке не будет превышать 10^{-3}:

Примеры

input.txt	output.txt
3 2 1 -1 8 -3 -1 2 -11 -2 1 2 -3	2.0000000000 3.0000000000 -1.0000000000
2 0.2 0.3 0.4 0.5 0.6 0.8	-0.0000000000 1.3333333333

Задача 8. Система сравнений по модулю

Источник:	основная II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	разумное
Ограничение по памяти:	разумное

Дана система сравнений по модулю простого числа P.

Если у системы существует единственное решение, нужно его вывести. В противном случае нужно вывести количество различных решений. Поскольку их может быть много, нужно найти количество по модулю $1\,000\,000\,007$.

Формат входных данных

В первой строке задано три целых числа: n — количество переменных, k — количество уравнений и P — модуль ($1 \leqslant n, k \leqslant 300$, $2 \leqslant P < 10^9$). Гарантируется, что число P — простое.

Следующие k строк описывают уравнения системы. В каждой строке $(n+1)$ целых чисел в диапазоне от 0 до $(P-1)$ включительно. Если обозначить i-ое из этих чисел через A_i, то уравнение выглядит так:

$$(A_1x_1 + A_2x_2 + A_3x_3 + \ldots + A_nx_n) \mod P = A_{n+1}$$

В уравнениях неизвестными переменными считаются x_1, x_2, …, x_n. В решении значения всех переменных должны быть в диапазоне от 0 до $(P-1)$ включительно.

Формат выходных данных

Если решение существует и единственно, то нужно его вывести. В этом случае выведите n целых чисел в диапазоне от 0 до $(P-1)$ — значения переменных x_1, x_2, …, x_n.

Если решения нет, выведите одно целое число ноль. Если решений много, выведите остаток от деления количества решений на (10^9+7).

Пример

input.txt	output.txt
4 4 7 1 2 3 4 5 0 1 4 0 1 0 0 1 1 2 0 0 0 1 6	1 1 3 3 6
3 2 2 1 1 0 1 1 1 0 1	4

Комментарий

Первый пример совпадает с первым тестом, а второй пример **не** совпадает со вторым.

Задача 9. Призрак Старого Парка

Источник:	повышенной сложности II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	разумное
Ограничение по памяти:	разумное

Предлагается написать программу для решения головоломки из старой компьютерной игры «Призрак Старого Парка». Посмотреть на головоломку в оригинале можно в этом видео.

Дано клеточное поле размера $N \times N$, в любой момент каждая клетка может быть белой или красной. Игрок может кликать мышкой на все клетки по собственному желанию. При нажатии на любую клетку в противоположный цвет перекрашиваются: эта клетка и все соседние с ней по стороне клетки. Обычно это пять клеток «крестом» вокруг той, на которую нажали, хотя около границы поля перекрашиваемых клеток может быть три или четыре.

Изначально все клетки поля белые. Во входном файле дана картинка, которую требуется получить. От вашей программы требуется вывести последовательность кликов, которая этого достигает.

Формат входных данных

В первой строке задано одно целое число N — размер поля ($1 \leqslant N \leqslant 20$).

В остальных N строках задано состояние поля, которое нужно получить. В каждой строке ровно N символов, каждый символ является одним из:

- точка (`'.'`, ASCII 46) — обозначает белый цвет,
- звёздочка (`'*'`, ASCII 42) — обозначает красный цвет.

Формат выходных данных

В первой строке нужно вывести одно целое число K — сколько кликов нужно сделать. Число K не должно превышать 10^6. В остальных K строках должны быть заданы координаты кликов в порядке их выполнения. Для каждого клика выведите целые числа R и C — номер строки и номер столбца для той клетки в таблице, куда нужно кликнуть ($1 \leqslant R, C \leqslant N$).

Гарантируется, что заданная головоломка разрешима вообще, и разрешима при заданном ограничении на количество кликов в частности.

Пример

input.txt	output.txt
5 ***** ***** ***** ***** *****	15 3 3 1 2 4 2 3 4 5 3 4 4 1 1 3 5 2 5 5 5 2 4 4 3 2 1 5 2 2 2
4 **.. ..*. **** .*..	4 2 2 4 4 4 3 2 1

Комментарий

Первый пример показывает, как решить оригинальную задачу из игры. Он совпадает с первым тестом. Второй пример **не** совпадает со вторым тестом.

Задача 10. Интерполяция

Источник:	повышенной сложности* II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	3 секунды*
Ограничение по памяти:	разумное

Одно из применений полиномиальной интерполяции — это замена трудновычислимой функции простым многочленом. Функция может вычисляться очень медленно, или вообще требовать несохраняемых данных для своего вычисления. Так, например, при пересечении двух криволинейных поверхностей в геометрическом моделировании кривую пересечения не сохраняют полностью, а сохраняют только результат интерполяции.

В этой задаче в качестве трудновычислимой функции будем использовать функцию:

$$f(x) = \sum_{i=1}^{N} w_i \cos(a_i x + b)$$

Параметры подобраны так, что эта функция гладкая и её значение и производные невелики.

Требуется вычислить функцию $f(x)$ в M заданных точках, используя заранее построенные интерполяционные многочлены. Необходимо разбить область определения на много отрезков, и на каждом отрезке посчитать свой интерполяционный многочлен невысокой степени. Точный подбор параметров (количество отрезков и степень) — это ваша задача.

Формат входных данных

В первой строке дано два целых числа: N — количество слагаемых в функции $f(x)$, M — количество точек, в которых нужно узнать значение функции ($1 \leqslant N \leqslant 10\,000$, $1 \leqslant M \leqslant 100\,000$).

В следующих N строках даны параметры слагаемых в функции. В каждой строке записано три вещественных числа w_i, a_i и b_i. При этом $|w_i| \leqslant 1$, $|a_i|, |b_i| \leqslant 20$.

В оставшихся M строках даны значения x, для которых нужно вычислить функцию $f(x)$. В каждой строке записано одно вещественное число в диапазоне $[0; 1]$.

Формат выходных данных

Нужно вывести ровно M вещественных чисел, по одному числу в строке. Рекомендуется выводить числа с максимальной точностью.

Абсолютная ошибка каждого вычисленного значения **не** должна превышать 10^{-12}.

Пример

input.txt	output.txt
2 5 0.76 1.0 0.5 -0.34 2.0 0.0 0.0 3.1415926536 2.6415926536 -0.5 1.0	0.32696274703668332107 -1.00696274703296428932 -0.94370278400100793270 0.57629721600483252431 0.19525019769348261889

Комментарий

Вы можете скачать второй тест по ссылке.

Задача 11. Параметры командной строки 3

Источник:	повышенной сложности II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	разумное
Ограничение по памяти:	разумное

Код задачи: `1/B_hardargs`

В программировании очень важно поддерживать совместимость между программами. Бывает необходимо поддержать поведение другой программы идеально точно, даже если это поведение кажется странным. В этой задаче предлагается реализовать алгоритм выделения параметров из командной строки в современной программе, собранной с помощью Microsoft Visual C.

Для простоты, командная строка будет содержать только следующие символы:
- латинские буквы (большие и маленькие) и цифры,
- двоеточие (ASCII 58) и точка (ASCII 46),
- пробел (ASCII 32),
- двойные кавычки (ASCII 34),
- обратный слэш (ASCII 92).

Правила разбора командной строки программой описаны в документации Microsoft. Следует заметить, что в реальном мире документация порой оказывается неполной или неверной. Скорее всего, вам понадобится дополнительная статья для полного понимания поведения.

Учтите, что вы можете очень легко самостоятельно проверить, какой ответ является правильным для любой заданной строки. Для этого создайте простую программу `test.c`, которая распечатывает содержимое массива `argv`, исключая нулевой элемент. Скомпилируйте программу с помощью Visual C, затем напишите в командной строке `text.exe`, пробел, строку из входных данных, и нажмите Enter.

ВАЖНО: программа должна быть собрана с помощью Visual C, т.к. в других компиляторах правила выделения параметров немного другие! Если у вас нет Visual Studio, можно скачать exeшник по этой ссылке. Он пишет параметры в файл output.txt, и должен работать не только в Windows, но даже в Linux + Wine.

Командная строка, которую необходимо разобрать, записана в единственной строке входного файла. Её длина не превышает 100 символов. Учтите, что в этой командной строке отсутствует имя исполняемого файла. В выходной файл нужно вывести искомые параметры командной строки, по одному параметру в строке. Каждый параметр должен быть заключён в квадратные скобки. Особый нулевой параметр (саму командную строку) выводить не нужно.

Данная задача проверяется на **emailtester** на четырёх разных компиляторах. Решение должно компилироваться и верно работать на всех четырёх компиляторах.

Примеры

input.txt	output.txt
"abc" d e	[abc] [d] [e]
a\\b d"e f"g h	[a\\b] [de fg] [h]
a\\\"b c d	[a\"b] [c] [d]

Задача 12. Система сравнений по модулю +

Источник:	космической сложности II
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	разумное

Дана система сравнений по модулю **произвольного** числа M.

Нужно вывести любое решение системы, если оно существует. Кроме того, нужно найти количество различных решений. Поскольку их может быть много, нужно найти количество по модулю $1\,000\,000\,007$.

Формат входных данных

В первой строке задано три целых числа: n — количество переменных, k — количество уравнений и M — модуль ($1 \leqslant n, k \leqslant 200$, $2 \leqslant M \leqslant 10^9$). Модуль M может **не** быть простым числом!

Следующие k строк описывают уравнения системы. В каждой строке $(n+1)$ целых чисел в диапазоне от 0 до $(M-1)$ включительно. Если обозначить i-ое из этих чисел через A_i, то уравнение выглядит так:

$$(A_1x_1 + A_2x_2 + A_3x_3 + \ldots + A_nx_n) \mod M = A_{n+1}$$

В уравнениях неизвестными переменными считаются x_1, x_2, …, x_n. В решении значения всех переменных должны быть в диапазоне от 0 до $(M-1)$ включительно.

Формат выходных данных

В первой строке выведите выведите остаток от деления количества решений на (10^9+7). Кроме того, требуется вывести решение, если хотя бы одно существует. В этом случае выведите n целых чисел в диапазоне от 0 до $(M-1)$ — значения переменных x_1, x_2, …, x_n.

Если решений много, можно вывести любое из них.

Пример

input.txt	output.txt
4 4 7 1 2 3 4 5 0 1 4 0 1 0 0 1 1 2 0 0 0 1 6	1 1 3 3 6
3 2 2 1 1 0 1 1 1 0 1	4 1 0 0

Комментарий

Первый пример совпадает с первым тестом, а второй пример **не** совпадает со вторым.