

## Задача А. Треугольник

Имя входного файла: `triangle.in`  
Имя выходного файла: `triangle.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Мальчик Вася очень любит треугольники, а именно прямоугольные. Как-то раз ему пришлось узнать минимальное расстояние от вершины при угле в 90 градусов до гипотенузы. Но Васе известны только длины катетов треугольника. Помогите ему.

### Формат входного файла

Входной файл содержит длины катетов — два целых числа, не превосходящие 1000.

### Формат выходного файла

Выходной файл должен содержать ответ к задаче с точностью три знака после десятичной точки.

### Примеры

<code>triangle.in</code>	<code>triangle.out</code>
3 4	2.400

## Задача В. Палиндромы

Имя входного файла: `palindromes.in`  
Имя выходного файла: `palindromes.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Однажды Вася пошел в школу и обнаружил на одной из стен слово. Возвращаясь со школы, он посмотрел на это же слово и прочитал его с другой стороны. Вася обнаружил, что оно читается одинаково с обеих сторон. Такие слова называются палиндромами. Например, *aba*, *a*, *abacaba* являются палиндромами, а *ab*, *baba*, *cab* — не являются. На следующей стене Вася заметил еще одно слово и задался вопросом: а на какое минимальное количество палиндромов можно его разбить?

Ваша задача — помочь Васе и написать программу, отвечающую на этот вопрос.

### Формат входного файла

Входной файл содержит единственную непустую строку, содержащую слово из строчных символов латинского алфавита (*a, b, c, ..., z*). Длина строки не превосходит 1000.

### Формат выходного файла

В первой строке выходного файла выведите минимальное количество палиндромов, на которые можно разбить слово. В следующей строке выведите сами палиндромы в порядке, в котором они находятся в слове. Палиндромы должны быть разделены точно одним пробелом.

### Примеры

<code>palindromes.in</code>	<code>palindromes.out</code>
<code>aba</code>	<code>1</code> <code>aba</code>
<code>abakada</code>	<code>3</code> <code>aba k ada</code>

## Задача С. Старый забор

Имя входного файла: `oldfence.in`  
Имя выходного файла: `oldfence.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Сому Тойеру необходимо покрасить старый забор, состоящий из  $n$  досок, в белый цвет, иначе он не сможет посидеть за компьютером. У Сомы Тойера есть несколько друзей, которые горят желанием помочь Сому. Друзья собрались вместе и распределили, кто какой участок забора будет красить. После покраски Сом заметил, что на некоторых участках забора находится несколько слоев краски, а некоторые доски так и остались непокрашенными.

Сом Тойер знает, кто из его друзей красил какой участок. Теперь ему нужно узнать, сколько краски ушло на различные участки этого забора. Сам он не может найти ответ. Помогите ему.

### Формат входного файла

Первая строка файла содержит длину забора  $n$  ( $1 \leq n \leq 1000000$ ), количество друзей  $m$  ( $1 \leq m \leq 200000$ ), и количество участков  $k$  ( $0 \leq k \leq 200000$ ), на которых Сом Тойер хочет узнать количество покрасок. Каждая доска забора нумеруется от 1 до  $n$ , друзья Сомы Тойера пронумерованы числами от 1 до  $m$ . В следующих  $m$  строках задаются участки, которые красил каждый из друзей Сомы. Участок задается двумя числами  $b_i$  и  $e_i$  — номерами досок ( $1 \leq b_i \leq e_i \leq n$ ) и количество слоев краски  $d_i$ , которые нанес друг на каждую из досок ( $0 \leq d_i \leq 1000000$ ). Следующие  $k$  строк содержат описание участков, для которых Сом хочет узнать количество потраченной краски (сумма количества слоев на каждой доске). Участок задается двумя числами  $c_i$  и  $d_i$  — номера досок.

### Формат выходного файла

Выходной файл должен содержать  $k$  строк, содержащих ответ на каждый вопрос Сомы (количество краски потраченной на каждом отрезке).

### Примеры

oldfence.in	oldfence.out
8 4 5	7
1 3 2	19
4 6 3	4
6 8 4	7
3 5 1	2
1 3	
5 8	
4 4	
3 4	
1 1	

## Задача D. Битва за жизнь

Имя входного файла: `dgamshuting.in`  
Имя выходного файла: `dgamshuting.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Несметное количество лет идёт война в стране Геймландия. Незаконным образом  $m$  равшанов отстраивают себе особнячки в виде башенок, состоящих из  $hp$  кирпичиков. Джемшуты это очень злит, потому раз в неделю собирается рать из  $r$  джемшуты и идет крушить эти башенки.

Каждый джемшут вооружён молотком, перфоратором и ломом. Также каждый джемшут имеет одну жизнь. У каждого равшана есть мегалазер на крыше башенки, который может уничтожить  $ar$  джемшуты за раз.

Так как живут равшаны и джемшуты в Геймландии, то их битва следует определенным правилам. Битва состоит из раундов. Каждый раунд состоит из двух действий:

1. каждый джемшут выбирает себе башенку и может в ней разобрать один кирпичик;
2. равшаны наносят ответный удар и уничтожают  $q \times ar$  джемшуты, где  $q$  — количество оставшихся башенок. Если все кирпичики в башенке разобраны, то она считается уничтоженной и равшан спасается бегством.

Требуется определить минимальное количество раундов, необходимое джемшутам для уничтожения всех башенок, или же установить, что джемшуты не смогут победить.

### Формат входного файла

Входной файл содержит четыре числа:  $d$  — количество джемшуты ( $1 \leq d \leq 10^6$ ),  $hp$  — количество кирпичиков в каждой башенке,  $ar$  — мощность мегалазера на крыше,  $r$  — количество равшанов ( $1 \leq hp, ar, r \leq 10^4$ ).

### Формат выходного файла

Выходной файл должен содержать одно число — минимальное количество раундов или  $-1$ , если джемшуты не смогут победить.

### Примеры

<code>dgamshuting.in</code>	<code>dgamshuting.out</code>
13 5 2 3	2
13 5 2 100	-1

## Задача Е. Рабочий стол

Имя входного файла: `desktop.in`  
Имя выходного файла: `desktop.out`  
Ограничение по времени: 1 секунда  
Ограничение по памяти: 256 мегабайт

Рабочий стол новой операционной системы ‘Winux’ представляет собой прямоугольник  $M \times N$  пикселей. Каждое окно в этой системе представляет собой прямоугольник, занимающий целое количество пикселей, со сторонами, параллельными границам рабочего стола.

Вася только что установил себе на компьютер эту систему и открыл  $K$  окон. И тут его любопытную натуру заинтересовал вопрос, какую часть рабочего стола он может увидеть. Иными словами, сколько пикселей рабочего стола не покрыто ни одним окном.

Сам он найти ответ не смог, зато кропотливо записал координаты левых верхних углов и размеры всех открытых окон. Теперь ваша задача — по этим данным найти ответ на поставленный вопрос.

### Формат входного файла

Первая строка входного файла содержит два целых положительных числа  $M, N$  — размеры рабочего стола в пикселях по вертикали и горизонтали соответственно ( $M \times N \leq 10000$ ). Во второй строке записано одно целое неотрицательное число  $K$  — количество открытых окон ( $K \leq 100000$ ).

Следующие  $K$  строк содержат описание окон. В каждой строке находятся по четыре целых числа  $(x_j, y_j, h_j, w_j)$ . Первые два числа описывают положение левого верхнего угла окна, а вторые два — его размеры по вертикали и горизонтали.

Координаты левого верхнего угла рабочего стола —  $(1, 1)$ , а правого нижнего —  $(M, N)$ . Гарантируется, что ни одно окно не выходит за границы рабочего стола, то есть  $(1 \leq x_j, x_j + h_j - 1 \leq M, 1 \leq y_j, y_j + w_j - 1 \leq N)$ . Окна могут накладываться и даже пересекаться.

### Формат выходного файла

В выходной файл выведите единственное число — количество пикселей рабочего стола, которые не закрыты ни одним окном.

### Примеры

desktop.in	desktop.out
5 8 2 1 1 3 2 5 1 1 7	27
4 3 2 2 2 2 2 3 1 1 2	7

## Задача F. Делимость

Имя входного файла: `divisibility.in`  
Имя выходного файла: `divisibility.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Дана строка с цифрами от 0 до 9. Используя эти цифры, нужно получить максимальное число, делящееся на 15. Цифры в этом числе не обязательно должны идти в том порядке, в котором они даны. Также, необязательно использовать все цифры. При этом, ведущие нули в получившемся числе запрещены.

### Формат входного файла

В первой строке находится число  $n$  — длина строки ( $1 \leq n \leq 10^6$ ). Во второй строке находится строка  $s$  длины  $n$  с цифрами от 0 до 9.

### Формат выходного файла

Выведите максимально возможное число, делящееся на 15, без ведущих нулей. Если такого числа получить нельзя, выведите в выходной файл `"impossible"` (без кавычек).

### Примеры

<code>divisibility.in</code>	<code>divisibility.out</code>
5 02041	4200
2 15	15

## Задача G. Хорошие билеты.

Имя входного файла: `tickets.in`  
Имя выходного файла: `tickets.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Билетик называется хорошим, если в его номере нет нулей, не более одной единицы, не более двух двоек, не более трёх троек, ..., не более девяти девяток.

При этом, цифры в номере билета должны идти в невозрастающем порядке. Ваша задача подсчитать, сколько существует различных хороших билетов, в номерах которых ровно  $N$  цифр.

### Формат входного файла

Во входном файле задано одно целое число  $N$  — количество цифр в билете ( $1 \leq N \leq 50$ ).

### Формат выходного файла

В выходной файл выведите единственное число — количество хороших билетов длины  $N$ .

### Примеры

<code>tickets.in</code>	<code>tickets.out</code>
5	1068

## Задача Н. Скоростная печать

Имя входного файла: `typeman.in`  
Имя выходного файла: `typeman.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

В стране Геймландия вышла новая игра, тренирующая остроту зрения. На экран выводится строка из маленьких латинских букв. Задача игрока — увидеть самую маленькую букву из встречающихся и назвать её. Но игра на этом не заканчивается. Далее начинается суперигра, в которой требуется найти самую маленькую букву, назвать её и количество раз, которое она встретилась в строке.

Одна буква меньше другой, если первая расположена в алфавите раньше. Требуется написать программу для проверки ответов участников.

### Формат входного файла

Входной файл содержит непустую строку  $s$ . Длина строки не превышает 200 символов. Строка состоит только из строчных букв латинского алфавита.

### Формат выходного файла

Выходной файл должен содержать одну букву — минимальный встретившийся в строке  $s$  символ и через пробел одно число — количество вхождений этого символа в строку.

### Примеры

<code>typeman.in</code>	<code>typeman.out</code>
abcdefgha	a 2



## Задача I. Ониме

Имя входного файла: `onime.in`  
Имя выходного файла: `onime.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

У программиста Васи есть  $X$  гигабайт ониме на жестком диске. Еще у него в наличии имеется бесконечное количество CD-дисков по 700 мегабайт каждый. Ему нужно записать всё ониме на диски. Какое минимальное число дисков ему при этом потребуется?

### Формат входного файла

В первой и единственной строке входного файла содержится число  $X$  ( $0 \leq X \leq 10^5$ ).

### Формат выходного файла

В единственной строке выходного файла выведите количество дисков, которое потребуется Васе.

### Примеры

<code>onime.in</code>	<code>onime.out</code>
1	2

## Задача J. Тренажерный зал

Имя входного файла: `gym.in`  
Имя выходного файла: `gym.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

В тренажерный зал пришли  $n$  программистов. После разминки тренер сказал им, что каждому из них сегодня необходимо сделать жим лежа общим количеством  $x$  кг. Конечно, сразу все  $x$  кг программист сделать не может, поэтому весь объем разбивается на небольшие веса (каждый программист может поднять штангу весом 20, 25, 30, 35 или 40 кг за один раз). Однако сами программисты не могут понять, как же разбить необходимые  $x$  кг на небольшие веса, так как в тренажерном зале нет компьютера, а телефоны отобрал злой тренер. Помогите им.

### Формат входного файла

В первой и единственной строке входного файла содержится число  $x$  — общий объем жима, который должны выполнить программисты ( $20 \leq x \leq 6000$ ). Гарантируется, что этот объем можно выполнить с помощью указанных весов.

### Формат выходного файла

Выведите последовательность весов, которые должен поднимать каждый программист. Сумма всех этих весов должна равняться итоговому объему в  $x$  килограммов. Разделяйте числа пробелами.

### Примеры

<code>gym.in</code>	<code>gym.out</code>
200	20 20 20 30 35 35 40

## Задача К. Лов Ктулху

Имя входного файла: `cthulhu.in`  
Имя выходного файла: `cthulhu.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

В середине прошлого века была объявлена Великая охота на Ктулху. Для этого сплели большую ктулхуловецкую сеть из серебряной проволоки. Она состоит из  $N \times M$  квадратных ячеек, границы которых состоят из единичных кусков проволоки.

Целая армада кораблей бороздила моря и океаны, стараясь разбудить чудовище. Но Ктулху так и не проснулся.

Прошло много лет, и сеть за ненадобностью решили переплавить, но перед этим нужно определить её вес. Из-за её больших размеров сделать это не так просто. Сможете ли вы определить, сколько весит сеть, если известно, что каждый единичный участок проволоки весит один грамм.

### Формат входного файла

Входной файл содержит два числа  $N, M$  — размеры сети ( $1 \leq N, M \leq 200$ )

### Формат выходного файла

Выходной файл содержит одно число — общий вес сети.

### Примеры

<code>cthulhu.in</code>	<code>cthulhu.out</code>
2 2	12