

## Задача A. Cars

Имя входного файла: `cars.in` или *стандартный ввод*  
Имя выходного файла: `cars.out` или *стандартный вывод*  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

...Зайдя в комнату Шерлока на Бейкер-Стрит, Ватсон сразу же заметил что-то неестественное. Во-первых, в комнате был наведён порядок. Во-вторых, к стене был прикреплен лист бумаги, на котором крупными буквами было написано. «Скрипка у меня. До связи. М.». Ватсон тут же позвонил Холмсу...

Звонок застал Шерлока в автомобиле... Одновременно пришло сообщение: «Карту вам передадут с курьером на Бейкер-Стрит. Ответьте с точностью до минуты, во сколько Вас ждать.». Как обычно, сообщение было отправлено с «утраченного» номера... пытаться вычислить отправителя бессмысленно — скорее всего, им окажется нанятый за пару фунтов по объявлению подросток... Остаётся принять игру. Тем более, что Бейкер-Стрит была уже совсем рядом. Если бы не пробка на узкой улице, которую осталось пересечь...

В пробке скопилось  $n$  автомобилей, стоящие один за другим; несколько студентов-волонтёров подсказывали водителям стоящих машин очередность движения, помогая автомобилям разъехаться. Холмс сразу определил, что это студенты-математики: каждую минуту автомобили, занимающие в пробке места с номерами, равными числам Фибоначчи, выбирались из пробки и уезжали (при этом в начале каждой минуты автомобили перенумеровываются слева направо, начиная с единицы). Новых автомобилей в пробке не появлялось.

Для того, чтобы ответить на сообщение похитителей, Холмсу надо подсчитать, через сколько минут все автомобили в пробке уедут.

### Формат входных данных

В единственной строке входного файла задано число  $n$  ( $1 \leq n \leq 10^{18}$ ) — количество автомобилей в пробке.

### Формат выходных данных

Выведите в выходной файл единственное число  $k$  — время (в минутах), которое потребуется, чтобы все автомобили в пробке разъехались.

### Примеры

<code>cars.in</code> или стандартный ввод	<code>cars.out</code> или стандартный вывод
2	1
5	2
12	3

## Задача B. Letter with Plan

Имя входного файла: `letter.in` или *стандартный ввод*  
Имя выходного файла: `letter.out` или *стандартный вывод*  
Ограничение по времени: 4 секунды  
Ограничение по памяти: 256 мегабайт

В переданное курьером письмо была вложена схема первого этажа одного из офисных центров Лондона. Холлы в офисном центре занумерованы числами от 1 до  $n$ , между некоторыми холлами проходят коридоры, причём между любыми двумя холлами можно пройти по цепочке коридоров. Двигаться по коридорам можно в обе стороны.

На обозначении каждого коридора шариковой ручкой была написана сумма номеров холлов, соединённых этим коридором. Сами номера были проставлены случайным образом — некоторые суммы явно были вычислены ошибочно.

Холмс сразу же догадался, что Мориарти делает «уводящий ход»: количество расстановок номеров может быть весьма велико... да и соответствует ли действительности данная карта — непонятно. А вот количество способов, которым можно восстановить номера, вполне может оказаться необходимым для разгадки ключом, задающим номер комнаты.

Требуется определить количество способов восстановить номера комнат (то есть количество различных перестановок чисел от 1 до  $n$ , при которых все обозначения на схеме будут соответствовать друг другу).

### Формат входных данных

В первой строке заданы целые числа  $n$  и  $m$  ( $1 \leq n \leq 10^5$ ,  $0 \leq m \leq 3 \cdot 10^5$ ) — количество комнат и коридоров соответственно. Далее в  $m$  строках заданы коридоры в формате  $a b s$  ( $1 \leq a, b \leq n$ ,  $1 \leq s \leq 2 \cdot n$ ), — номера (в соответствии со схемой) холлов, соединяемых коридором дороги и сумма, написанная на обозначении коридора.

### Формат выходных данных

В первой строке выведите  $k$  — количество способов восстановить номера комнат. Если  $k > 0$ , во второй строке выведите один любой способ — перестановку из чисел от 1 до  $n$ , где  $i$ -е число означает «восстановленный» номер комнаты с исходным номером  $i$ . Обратите внимание, что автор схемы вполне мог сделать так, что восстановление номеров комнат может оказаться невозможным.

### Примеры

<code>letter.in</code> или стандартный ввод	<code>letter.out</code> или стандартный вывод
3 2 2 1 3 3 1 4	1 1 2 3
2 1 1 2 3	2 1 2

## Задача C. Long Queue

Имя входного файла:	<code>longqueue.in</code> или <i>стандартный ввод</i>
Имя выходного файла:	<code>longqueue.out</code> или <i>стандартный вывод</i>
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Чтобы добраться до указанного в письме офисного центра, Холмс и Ватсон решили воспользоваться такси. Вежливый голос диспетчера сообщил, что в связи с «русской забастовкой водителей» заказы не принимаются, и договариваться придётся на стоянке такси.

Около стоянки собралась толпа пассажиров, о чём-то ругающихся с водителями. Назревала драка. Благодаря армейскому опыту Ватсона и боксёрским навыкам Холмса драку удалось предотвратить. Суть «русской забастовки» заключалась в следующем: таксисты соглашались возить пассажиров строго по одному, следуя при этом только по определённым маршрутам (каждый по своему). Для каждого пассажира известен список номеров такси, маршрут которых его устраивает; если маршрут пассажира не устраивает, вариант движения «с пересадками» для него недопустим.

Тем временем пассажиры выстроились в очередь, получив номера от 1 до  $n$ . Ватсон решил организовать посадку пассажиров. Сначала Ватсон находит машину для первого пассажира в очереди, тот уезжает, затем наступает очередь второго и так далее. При этом может случиться ситуация, в которой некоторому пассажиру машину подобрать не удастся и уехать тот не сможет.

Пока Ватсон обдумывал оптимальный вариант организации посадки, Холмс успел заметить среди собравшихся некоторое количество агентов Лестрейда. Они явно собирались последовать за Холмсом, надеясь, что тот приведёт их к логову Мориарти. Рисковать скрипкой Холмс не хотел, так что он предложил Ватсону действовать таким образом, чтобы ни один из агентов не смог уехать.

Сможет ли Ватсон таким образом нейтрализовать всех агентов?

### Формат входных данных

Первая строка входного файла содержит 2 целых числа  $n$  и  $k$  ( $2 \leq n, k \leq 50$ ) — количество пассажиров и количество такси соответственно. Каждая из следующих  $n$  строк содержит список «подходящих» номеров для  $i$ -го в порядке очереди пассажира: количество машин  $m_i$  ( $1 \leq m_i \leq k$ ) и  $m_i$  целых чисел от 1 до  $k$  — количество и номера «подходящих» машин в порядке возрастания (такси занумерованы целыми положительными числами от 1 до  $k$ ).

Следующая строка содержит количество запросов  $q$  ( $1 \leq q \leq 1000$ ). Каждая из следующих  $q$  строк содержит один запрос и имеет следующий формат:  $t_j$  — количество агентов ( $1 \leq t_j \leq n - 1$ ) и  $t_j$  целых чисел от 1 до  $n$  — количество и номера полицейских агентов в порядке возрастания. Запросы обрабатываются независимо друг от друга (то есть в каждом запросе полицейскими агентами являются те и только те пассажиры, номера которых заданы в запросе)

### Формат выходных данных

Для каждого запроса выведите единственную строку, содержащую «Yes», если Ватсон сможет «отсечь» всех полицейских агентов, и «No» в противном случае.

## Примеры

longqueue.in или стандартный ввод	longqueue.out или стандартный вывод
3 3 2 2 3 2 1 2 2 2 3 3 1 1 1 2 1 3	No No Yes
3 3 2 2 3 2 2 3 2 1 2 3 1 1 1 2 1 3	No No No
4 4 4 1 2 3 4 4 1 2 3 4 2 1 2 2 1 2 4 1 2 1 3 1 4 2 3 4	No Yes Yes Yes

## Задача D. Minecraft

Имя входного файла:	<code>minecraft.in</code> или <i>стандартный ввод</i>
Имя выходного файла:	<code>minecraft.out</code> или <i>стандартный вывод</i>
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Офис с вычисленным из записки Мориарти номером неожиданно оказался в подвале офисного центра. Надпись на двери гласила «Minecraft Research Laboratory, project MinecraftReality». Холмс вспомнил всё, что он читал про это. «Первый удачный эксперимент британских учёных по воссозданию игрового мира в реальности», «Игры выходят из экрана» и так далее. Мориарти вполне мог быть причастен к созданию математической модели этого проекта. Да, похоже, что сейчас придётся встретиться с технологиями будущего, использованными для воссоздания компьютерной игры прошлого...

Вид сбоку игрового мира Minecraft представляет собой прямоугольное поле размером  $h \times w$  — вертикальный разрез земли. Каждая клетка может быть пустой, занятой блоком земли или занятой блоком с ключом. Под нижней границей поля находится лава. Под каждым блоком находится либо другой блок, либо лава. Кроме того, в мире Minecraft действует гравитация, поэтому пока у объекта нет опоры, он падает вниз.

Изначально игрок выходит на поле в пустом блоке, находящемся над самым левым верхним занятым.

Возможные действия игрока:

1. Выкопать блок снизу, если под этим блоком нет лавы. После этого игрок перемещается на одну клетку вниз.
2. Выкопать блок справа. Если этот блок не самый верхний в столбце, то блоки, находившиеся выше, падают вниз. Игрок вправо не перемещается.
3. Создать блок земли в пустой клетке справа. Если под этим блоком находится пустая клетка, то блок падает вниз.
4. Сделать шаг в пустую клетку справа. Если под игроком окажется пустая клетка, то он падает вниз. Игрок может безопасно падать с любой высоты.
5. Если игрок не находится у верхней границы, он может прыгнуть на одну клетку вверх. Находясь в прыжке, можно создать блок земли под собой или выполнить одно из вышеперечисленных действий.

Цель игрока — выкопать все блоки с ключами и добраться до правого края карты (выхода из Minecraft Reality) за минимальное количество операций создания и удаления блоков. Обратите внимание, что выкапывание блока с ключом тоже является операцией удаления блока!

### Формат входных данных

Первая строка входного файла содержит два целых числа  $h$  и  $w$  ( $5 \leq h, w \leq 100$ ) — высота и ширина поля соответственно. Следующие  $h$  строк по  $w$  символов каждая описывают поле: символ «.» обозначает пустую клетку, символ «#» обозначает блок земли, символ «G» обозначает блок с ключами. У левой и правой границ блоков с ключами нет. Все клетки у верхней границы пустые. Все клетки у нижней границы заняты блоками земли. Ни под каким занятым блоком не находится пустой блок.

### Формат выходных данных

Выходной файл должен содержать одно число — минимальное количество операций создания и удаления блоков, необходимых для достижения цели.

## Примеры

minecraft.in или стандартный ввод	minecraft.out или стандартный вывод
5 5 ..... #..#. ##.## #GGG# #####	5
5 5 ..... #..G. #..## ##### #####	1
8 5 ..... ..... ....# ....# ....# .#.G# ##### #####	4

## Задача E. Safe

Имя входного файла:	<code>safe.in</code> или <i>стандартный ввод</i>
Имя выходного файла:	<code>safe.out</code> или <i>стандартный вывод</i>
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Итак, все ключи подошли. Вот и нужная комната... в её центре стоит массивное сооружение. Это сейф с компьютерным замком — похоже, последняя (на сегодня) задачка профессора Мориарти. Остаётся только вскрыть замок.

Интерфейс замка представляет собой своеобразный пасьянс: в середине экрана записано число 1; также дана колода карточек вида « $s\ n$ », где  $s$  — это одна из операций «+», «-», «\*» или «/», а  $n$  — это цифра от 1 до 9.

Холмс должен положить эти карточки на единицу в таком порядке, чтобы при последовательном применении написанных на них операций к 1 получилось наибольшее возможное значение.

Для контроля корректности найденного решения (при неправильном вводе кода содержимое сейфа уничтожается) брат Холмса Майкрофт попросил Вас написать программу, располагающую карточки в нужном порядке автоматически.

### Формат входных данных

Первая строка входного файла содержит одно целое число  $N$  ( $1 \leq N \leq 100$ ) — количество карточек в наборе. Каждая из следующих  $N$  строк содержит описание карточки: разделённые пробелом знак операции и число, написанные на соответствующей карточке.

### Формат выходных данных

Выходной файл должен содержать  $N$  строк — описания карточек в том же формате, перечисленные в порядке их применения, начиная с первой применяемой. Если решений несколько, выведите любое из них.

### Примеры

<code>safe.in</code> или стандартный ввод	<code>safe.out</code> или стандартный вывод
8	/ 2
+ 1	/ 3
+ 5	+ 1
- 1	+ 5
- 2	* 2
* 2	* 4
* 4	- 1
/ 2	- 2
/ 3	

## Задача F. Serpinski Carpet

Имя входного файла: `serpinski.in` или *стандартный ввод*  
Имя выходного файла: `serpinski.out` или *стандартный вывод*  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Вернувшись со скрипкой домой, Шерлок Холмс обнаружил очередное письмо. Текст письма гласил:

«Мистер Холмс! Несколько лет назад Вы посещали прочитанные мной публичные лекции по дискретной математике. Исходя из интересов дела, в то время мне было удобнее действовать под видом американского профессора. Я был очень раздосадован тем, что столь талантливый студент так и не удосужился пройти финальное испытание, и поэтому решил, воспользовавшись столкновением наших интересов, провести его для Вас сегодня. Вы успешно справились со всеми поставленными мной задачами, так что в этом конверте Вы найдёте официальный сертификат на Ваше имя. Профессор Д.М.».

Теперь, когда Холмсу нечего делать, он достаёт оружие... вспоминает, что является обладателем отличной оценки по математике... и начинает стрелять «на точность» по ковру Серпинского.

«Ковёр Серпинского» — один из самых известных фракталов — строится следующим образом. На первом шаге имеем квадрат с противоположными вершинами в точках  $(0, 0)$  и  $(L, L)$ . Делим квадрат на девять квадратов со стороной  $L/3$ , после чего выбрасываем внутреннюю часть среднего квадрата (оставляя границу). Далее с каждым из оставшихся квадратов продельвается та же операция и так далее. Получается объект с бесконечным количеством дырок.

Ваша задача — найти расстояние от точки попадания пули до ковра Серпинского.

### Формат входных данных

Входной файл состоит из не более 50 тестовых примеров. Первая строка каждого тестового примера содержит целое число  $L$  ( $1 \leq L \leq 1000$ ) — длину стороны первоначального квадрата. Следующая строка содержит два вещественных числа — координаты точки попадания пули. Гарантируется, что пуля всегда попадает в первоначальный квадрат.

Входной файл завершается примером с  $L = 0$ , обрабатывать который не требуется.

### Формат выходных данных

Выведите одно число — расстояние от точки попадания пули до ковра Серпинского со стороной  $L$  с точностью не хуже  $10^{-6}$ .

### Пример

<code>serpinski.in</code> или стандартный ввод	<code>serpinski.out</code> или стандартный вывод
1	0.066667
0.4 0.4	0.000000
81	0.000000
18 39.825	3.333333
60	
30 15	
60	
30 10	
0	