

## Задача А. АвиAPERелеты

Имя входного файла: `avia.in`  
Имя выходного файла: `avia.out`  
Ограничение по времени: 1 секунда (Python: 2 с)  
Ограничение по памяти: 64 мегабайта

Профессору Форду необходимо попасть на международную конференцию. Он хочет потратить на дорогу наименьшее количество денег, поэтому решил, что будет путешествовать исключительно ночными авиарейсами (чтобы не тратиться на ночевку в отелях), а днем будет осматривать достопримечательности тех городов, через которые он будет проезжать транзитом. Он внимательно изучил расписание авиAPERелетов и выбрал подходящие ему авиарейсы, выяснив, что перелеты на выбранных направлениях совершаются каждую ночь и за одну ночь он не сможет совершить два перелета.

Теперь профессор хочет найти путь наименьшей стоимости, учитывая что до конференции осталось  $K$  ночей (то есть профессор может совершить не более  $K$  перелетов).

### Формат входных данных

В первой строке находятся числа  $N$  (количество городов),  $M$  (количество авиарейсов),  $K$  (количество оставшихся ночей),  $S$  (номер города, в котором живет профессор),  $F$  (номер города, в котором проводится конференция). Ограничения:  $2 \leq N \leq 100$ ,  $1 \leq M \leq 10^5$ ,  $1 \leq K \leq 100$ ,  $1 \leq S \leq N$ ,  $1 \leq F \leq N$ .

Далее идет  $M$  строк, задающих расписание авиарейсов.  $i$ -я строка содержит три натуральных числа:  $S_i$ ,  $F_i$  и  $P_i$ , где  $S_i$  — номер города, из которого вылетает  $i$ -й рейс,  $F_i$  — номер города, в который прилетает  $i$ -й рейс,  $P_i$  — стоимость перелета  $i$ -м рейсом.  $1 \leq S_i \leq N$ ,  $1 \leq F_i \leq N$ ,  $1 \leq P_i \leq 10^6$ .

### Формат выходных данных

Выведите одно число — минимальную стоимость пути, подходящего для профессора. Если профессор не сможет за  $K$  ночей добраться до конференции, выведите число -1.

### Пример

<code>avia.in</code>	<code>avia.out</code>
4 5 2 1 4 1 2 1 2 3 1 3 4 1 1 3 3 1 4 5	4

## Задача В. Опасный маршрут

Имя входного файла: `danger.in`  
Имя выходного файла: `danger.out`  
Ограничение по времени: 1 секунда  
Ограничение по памяти: 64 мегабайта

Профессор Дейкстра живёт в очень опасном районе города. Ежедневно бандиты грабят на улицах прохожих. Читая криминальную хронику, профессор вычислил вероятность быть ограбленным при проходе по каждой улице города.

Теперь он хочет найти наиболее безопасный путь от дома до университета, в котором он преподаёт. Иными словами, он хочет найти путь от дома до университета, для которого вероятность быть ограбленным минимальна.

### Формат входных данных

В первой строке входного файла записаны два числа  $N$  и  $M$  — количество зданий и количество улиц, соединяющих здания ( $1 \leq N \leq 100$ ,  $1 \leq M \leq N(N-1)/2$ ). В следующей строке находятся числа  $S$  и  $F$  — номер дома, в котором живёт профессор и номер дома, в котором находится университет соответственно. Далее в  $M$  строках расположены описания дорог: 3 целых числа  $S_i$ ,  $F_i$  и  $P_i$  — номера зданий, в которых начинается и заканчивается дорога, и вероятность в процентах быть ограбленным, пройдя по дороге соответственно ( $1 \leq S_i \leq N$ ,  $1 \leq F_i \leq N$ ,  $0 \leq P_i \leq 100$ , дороги двунаправленные). Гарантируется, что существует хотя бы один путь от дома профессора до университета.

### Формат выходных данных

Необходимо вывести одно число — минимальную возможную вероятность быть ограбленным с точностью не менее 6 знаков после запятой.

### Пример

<code>danger.in</code>	<code>danger.out</code>
3 3 1 3 1 2 20 1 3 50 2 3 20	0.36

### Замечания

Для приведенного примера минимальная вероятность быть ограбленным достигается на маршруте 1 – 2 – 3.

Указание. Пусть вероятность быть ограбленным на пути  $A-B$  равна  $p$ , а на пути от  $B-C$  равна  $q$ . Какова вероятность быть ограбленным на пути  $A-B-C$ ?

## Задача С. Транспортировка

Имя входного файла: `cups.in`  
Имя выходного файла: `cups.out`  
Ограничение по времени: 1 секунда (Python: 4 с)  
Ограничение по памяти: 64 мегабайта

Компания “Яндекс” решила подарить всем школьникам и преподавателям ЛКШ оригинальные кружки. К сожалению, количество необходимых кружек оказалось столь велико, что изготовитель доставил кружки в офис Яндекса в самый последний момент. На то, чтобы довести кружки из Москвы до “Берендеевых полей” осталось всего 24 часа.

Заказ на  $10^7$  экземпляров кружек (а именно столько заказали организаторы), конечно же, за один рейс не увезти. Однако, за первый рейс хочется привезти максимальное количество кружек. Для перевозки был заказан один большегрузный автомобиль. Но есть один нюанс: на некоторых дорогах установлено ограничение на вес автомобиля. Поэтому если автомобиль нагрузить кружками под завязку, то, возможно, не удастся воспользоваться самым коротким маршрутом, а придётся ехать в объезд. Может случиться даже так, что из-за этого грузовик не успеет доехать до лагеря вовремя, а этого допустить никак нельзя.

Итак, сколько же кружек можно погрузить в автомобиль, чтобы успеть привезти этот ценный груз вовремя и не нарушить правила дорожного движения?

### Формат входных данных

В первой строке находятся числа  $n$  ( $1 \leq n \leq 500$ ) и  $m$  — количество узловых пунктов дорожной схемы и количество дорог, соответственно. В следующих  $m$  строках находится информация о дорогах. Каждая дорога описывается в отдельной строке следующим образом. Сначала указаны номера узловых пунктов, которые соединяются данной дорогой, потом время, которое тратится на проезд по этой дороге, и, наконец, максимальный вес автомобиля, которому разрешено ехать по этой дороге. Известно, что все дороги соединяют различные пункты, причем для каждой пары пунктов есть не более одной дороги, непосредственно их соединяющей. Все числа разделены одним или несколькими пробелами.

Узловые пункты нумеруются числами от 1 до  $n$ . При этом офис Яндекса имеет номер 1, а “Берендеевы поляны” — номер  $n$ . Время проезда по дороге задано в минутах и не превосходит 1440 (24 часа). Ограничение на массу задано в граммах и не превосходит одного миллиарда. Кроме того, известно, что одна кружка весит 100 грамм, а пустой грузовик — 3 тонны.

### Формат выходных данных

Выведите одно число — максимальное количество кружек, которое можно привезти за первый рейс, потратив не более 24 часов.

### Пример

<code>cups.in</code>	<code>cups.out</code>
3 3	2
1 2 10 3000220	
2 3 20 3000201	
1 3 1 3000099	

## Задача D. Наименьшее кратное

Имя входного файла: `multiple.in`  
Имя выходного файла: `multiple.out`  
Ограничение по времени: 1 секунда (Python: 2 с)  
Ограничение по памяти: 64 мегабайта

Дано число  $X$  и множество цифр  $D$ . Требуется дописать к  $X$  минимальное количество цифр из  $D$ , чтобы получившееся число делилось на  $k$ . При этом получившееся число должно быть минимально возможным.

### Формат входных данных

Первая строка входного файла содержит два натуральных числа  $X$  и  $k$  ( $1 \leq X \leq 10^{1000}, 2 \leq k \leq 100\,000$ ). Во второй строке записано количество цифр во множестве  $D$ . В третьей строке через пробел записаны эти цифры.

### Формат выходных данных

Единственная строка должна содержать минимальное число, полученное из  $X$  дописыванием цифр из  $D$  и кратное  $k$ . Если такого числа не существует, выведите `-1`.

### Пример

<code>multiple.in</code>	<code>multiple.out</code>
102 101 3 1 0 3	10201
202 101 3 1 0 3	202

## Задача E. Pink Floyd

Имя входного файла: `floyd.in`  
Имя выходного файла: `floyd.out`  
Ограничение по времени: 1 секунда (Python: 4 с)  
Ограничение по памяти: 64 мегабайта

Группа *Pink Floyd* собирается отправиться в новый концертный тур по всему миру. По предыдущему опыту группа знает, что солист *Роджер Уотерс* постоянно нервничает при перелетах. На некоторых маршрутах он теряет вес от волнения, а на других — много ест и набирает вес.

Известно, что чем больше весит Роджер, тем лучше выступает группа, поэтому требуется спланировать перелеты так, чтобы вес Роджера на каждом концерте был максимально возможным.

Группа должна посещать города в том же порядке, в котором она дает концерты. При этом между концертами группа может посещать промежуточные города.

### Формат входных данных

Первая строка входного файла содержит три натуральных числа  $n$ ,  $m$  и  $k$  — количество городов в мире, количество рейсов и количество концертов, которые должна дать группа соответственно ( $n \leq 100$ ,  $m \leq 10\,000$ ,  $2 \leq k \leq 10\,000$ ). Города пронумерованы числами от 1 до  $n$ .

Следующие  $m$  строк содержат описание рейсов, по одному на строке. Рейс номер  $i$  описывается тремя числами  $b_i$ ,  $e_i$  и  $w_i$  — номер начального и конечного города рейса и предполагаемое изменение веса Роджера в миллиграммах ( $1 \leq b_i, e_i \leq n$ ,  $-100\,000 \leq w_i \leq 100\,000$ ).

Последняя строка содержит числа  $a_1, a_2, \dots, a_k$  — номера городов, в которых проводятся концерты ( $a_i \neq a_{i+1}$ ). В начале концертного тура группа находится в городе  $a_1$ .

Гарантируется, что группа может дать все концерты.

### Формат выходных данных

Первая строка выходного файла должна содержать число  $l$  — количество рейсов, которые должна сделать группа. Вторая строка должна содержать  $l$  чисел — номера используемых рейсов.

Если существует такая последовательность маршрутов между концертами, что Роджер будет набирать вес неограниченно, то первая строка выходного файла должна содержать строку `infinitely kind`.

### Пример

floyd.in	floyd.out
4 8 5 1 2 -2 2 3 3 3 4 -5 4 1 3 1 3 2 3 1 -2 3 2 -3 2 4 -10 1 3 1 2 4	6 5 6 5 7 2 3
4 8 5 1 2 -2 2 3 3 3 4 -5 4 1 3 1 3 2 3 1 -2 3 2 -3 2 4 10 1 3 1 2 4	infinitely kind

## Замечания

Вместо python3 для сдачи этой задачи используйте руру3.

## Задача F. Вслепую по лабиринту

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

### Интерактивная задача

Ваша цель — написать программу, управляющую роботом, идущим вслепую по лабиринту. Лабиринт состоит из  $N \times M$  ( $1 \leq N, M \leq 30$ ) клеток. Каждая из клеток может быть свободной или заблокированной (непроходимой). Все клетки на границе лабиринта непроходимые. Робот начинает работу в свободной клетке, он может переместиться на юг, запад, север или восток в свободную клетку. Робот не имеет оптических сенсоров, только сенсор удара, так что при попытке перемещения в заблокированную клетку сработает сенсор и робот останется в той же клетке. При этом вы получаете сообщение о том, что клетка заблокирована. Робот должен побывать во всех проходимых клетках лабиринта, достижимых из начальной клетки.

### Протокол интерактивного взаимодействия

Программа должна выводить на стандартный вывод одну строку с действием робота и ждать строки в стандартном вводе с ответом, затем выводить очередную строку с действием и считывать ответ и так далее до тех пор, пока все проходимые клетки лабиринта не будут посещены. Программа должна завершать работу только когда все проходимые клетки будут посещены. Проходимые клетки могут быть посещены несколько раз. Допустимо передвигаться даже если все проходимые клетки уже посещены.

### Формат выходных данных

Каждая строка выходных данных должна представлять собой команду для робота. Это должна быть одна из пяти возможных строк: SOUTH, WEST, NORTH, EAST или DONE. Строка DONE должна быть напечатана после посещения роботом всех проходимых клеток. После вывода DONE программа должна завершать свою работу. Вывод каждой команды обязательно должен завершаться выводом конца строки.

### Формат входных данных

Каждая строка входных данных представляет собой ответ на действие робота. Это может быть строка EMPTY если робот успешно переместился в заданном направлении или строка BLOCKED если робот не смог переместиться из-за того, что клетка, в которую он хотел попасть, непроходима.

### Пример

стандартный ввод	стандартный вывод
NORTH	BLOCKED
EAST	BLOCKED
SOUTH	EMPTY
EAST	BLOCKED
SOUTH	BLOCKED
WEST	EMPTY
SOUTH	BLOCKED
WEST	BLOCKED
NORTH	EMPTY
WEST	EMPTY
WEST	BLOCKED
NORTH	BLOCKED
EAST	EMPTY
NORTH	BLOCKED
DONE	

Иллюстрация к примеру:

