Задача A. Average Convex Hull

Имя входного файла:	`average.in`
Имя выходного файла:	`average.out`
Ограничение по времени:	6 seconds
Ограничение по памяти:	256 мебибайт

Выпуклой оболочкой множества S на плоскости называется минимальный выпуклый многоугольник, который содержит все точки, принадлежащие множеству S.

Вам заданы n точек на плоскости. Одна из точек выбирается случайным образом и удаляется из множества, при этом для каждой точки вероятность того, что будет удалена именно эта точка, одинакова.

Найдите математическое ожидание количества вершин выпуклой оболочки получившегося множества. При этом если выпуклой оболочкой является отрезок, то считается, что у выпуклой оболочки две вершины. Если выпуклой оболочкой является невырожденный многоугольник, то углы в каждой из его вершин строго меньше π.

Формат входного файла

В первой строке входного файла содержится целое число n — количество заданных точек ($3 \leqslant n \leqslant 200\,000$). Каждая из последующих n строк содержит по два целых числа, по модулю не превосходящих 10^9: i-я строка содержит координаты i-й точки. При этом все n точек попарно различны.

Формат выходного файла

Выведите математическое ожидание количества вершин выпуклой оболочки множества, получившегося после удаления точки, в виде несократимой дроби p/q.

Примеры

average.in	average.out
5 0 0 0 4 4 0 3 3 4 4	17/5

Задача B. Binary Suffix Array

Имя входного файла:	`binary.in`
Имя выходного файла:	`binary.out`
Ограничение по времени:	6 seconds
Ограничение по памяти:	256 мебибайт

Двоичным словом называется слово, состоящее только из нулей и единиц.

Рассмотрим двоичное слово w, длина которого равна n. Суффиксным массивом слова w называется такой массив $a[1..n]$, что $w[a[i]..n]$ является i-м лексикографически суффиксом среди всех суффиксов слова w. Например, если w=“`001011`”, лексикографический порядок его суффиксов таков: “`001011`”, “`01011`”, “`011`”, “`1`”, “`1011`”, “`11`”, и суффиксным массивом слова w будет массив $(1, 2, 4, 6, 3, 5)$.

Вам задан суффиксный массив некоторого двоичного слова w. Требуется восстановить w.

Формат входного файла

В первой строке входного файла задано одно целое число n — длина слова w ($1 \leqslant n \leqslant 300\,000$). Вторая строка содержит n попарно различных целых положительных чисел, не превосходящих n — суффиксный массив слова w.

Формат выходного файла

Выведите двоичное слово w такое, что заданный во входном файле массив является суффиксным массивом для этого слова. Если таких слов несколько, выведите любое. Если таких слов не существует, выведите “`Error`”.

Примеры

binary.in	binary.out
6 1 2 4 6 3 5	001011

Задача C. Collision Detection (Division 1 Only!)

Имя входного файла:	`collision.in`
Имя выходного файла:	`collision.out`
Ограничение по времени:	5 seconds
Ограничение по памяти:	256 мебибайт

Компания *Giggle* разрабатывает автомобиль, который сможет ездить по дорогам автоматически, без управления человеком. На данный момент построен его прототип, который может двигаться по прямой. Очередная задача разработчиков — проверить работу системы предотвращения столкновений. Для этого они планируют провести следующий эксперимент.

В эксперименте задействованы n автомобилей. Изначально автомобили располагаются на заданных позициях, при этом каждый автомобиль направлен или на север, или на юг, или на запад, или на восток. После этого автомобили начинают двигаться в соответствующем направлении со скоростью 1 метр в секунду. Система должна определить ситуации, в которых возможны столкновения, и предотвратить их.

Для того, чтобы вычислить ожидаемое поведение автомобилей на трассе, инженерам компании Giggle необходимо знать, для каких двух автомобилей опасность столкновения максимальна. В качестве величины, определяющей опасность столкновения между двумя автомобилями используется минимальное расстояние между автомобилями после начала движения: чем это расстояние меньше, тем выше опасность столкновения. В рамках данной задачи автомобили считаются точками, а расстояние между ними измеряется как длина отрезка, соединяющего эти точки.

Формат входного файла

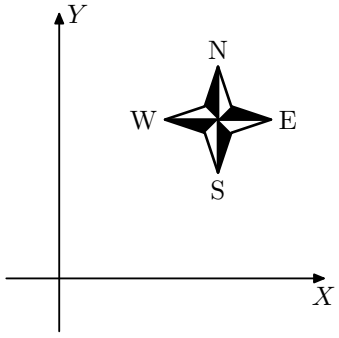
Входной файл состоит из нескольких тестовых примеров.

Первая строка каждого тестового примера содержит n — количество автомобилей, участвующих в эксперименте ($2 \leqslant n \leqslant 100\,000$).

Последующие n строк содержат по два целых числа x_i и y_i, за которыми следует один символ c_i. Каждая такая строка задаёт начальное положение автомобиля и направление, в котором он едет. Координаты заданы в метрах и не превосходят по модулю 10^8. Направление задаётся одним из следующих символов:

“`N`” — север, “`S`” — юг, “`E`” — восток, “`W`” — запад.

При этом начальное положение всех автомобилей попарно различно.



За последним тестовым примером следует тестовый пример с $n = 0$, обозначающий конец входного файла. Этот пример обрабатывать не нужно. Сумма значений n во всех тестовых примерах не превосходит 100 000.

Формат выходного файла

Для каждого тестового примера выведите по три строки. Первая строка должна содержать число d — минимальное расстояние в метрах между некоторыми двумя автомобилями с момента начала движения. Во вторую строку выведите два целых числа a и b — номера автомобилей, которые в

какой-то момент после начала движения окажутся на расстоянии d. Автомобили занумерованы, начиная с единицы, в порядке, в котором они заданы во входном файле. Третья строка должна содержать время t от начала движения, выраженное в секундах — момент времени, когда автомобили с номерами a и b окажутся на расстоянии d. Если возможны несколько правильных ответов, выведите любой из них.

Ответы на два соседних тестовых примера разделяйте пустой строкой.

Проверяющая программа производит все сравнения вещественных чисел с абсолютной или относительной точностью 10^{-6}.

Примеры

collision.in	collision.out
4 0 0 E 3 3 N 0 4 S 4 0 E 4 0 0 S 3 3 N 0 4 S 4 0 E 2 0 0 S 0 1 N 0	2.8284271247461903 1 3 2.0 3.0 2 3 0.5 1.0 2 1 0.0

Задача D. Dual Cure (Division 1 Only!)

Имя входного файла:	`dual.in`
Имя выходного файла:	`dual.out`
Ограничение по времени:	4 seconds
Ограничение по памяти:	256 мебибайт

Среди населения Северо-Восточной Антаркики вспыхнула эпидемия носорожьего гриппа. Вирус носорожьего гриппа склонен к мутациям, что существенно осложняет лечение заболевания. К счастью, врачи разработали подход, который позволяет эффективно лечить носорожий грипп в случае, если известен источник инфекуии.

На данный момент в госпитале находятся n больных носорожьим гриппом. Для каждого больного известно, от кого он заразился. Некоторые больные заразились от носорогов, некоторые — от других больных.

Лечением пациентов занимаются два врача. Излечение одного пациента, включающее определение конкретной мутации вируса, занимает 1 час. Пациент, заразившийся от носорога, может быть вылечен в любой момент. Для того, чтобы вылечить пациента X, заразившегося от пациента Y, врач должен знать, какая именно мутация вируса была обнаружена пациента Y, таким образом, Y должен быть вылечен ранее, чем X. Передача информации от одного врача к другому (в силу действующих в госпитале формальностей) занимает также один час, то есть в случае, если пациент X обратился не к тому же доктору, что и пациент Y, то он может быть принят только через час после того, как был вылечен Y.

Ваша задача — составить график приёма пациентов каждым из докторов таким образом, чтобы последний из пациентов был вылечен как можно ранее.

Формат входного файла

В первой строке входного файла задано целое число n — количество пациентов ($1 \leqslant n \leqslant 100\,000$). Пациенты пронумерованы от 1 до n. Следующая строка содержит n целых чисел $a_1, a_2, \ldots, a_n$. Для каждого i от 1 до n число a_i обозначает номер пациента, от которого заразился пациент i, или 0, если он заразился от носорога ($a_i < i$).

Формат выходного файла

В первой строке выходного файла выведите целое число t — наименьшее количество часов, через которое все пациенты будут вылечены. Каждая из последующих t строк содержит по два целых числа. В i-й из этих строк содержатся числа u_i и v_i — номера пациентов, принимаемых в течение i-го часа первым и вторым доктором соответственно. Если какой-либо доктор в течении некоторого часа свободен, соответствующее число должно быть равно нулю.

Примеры

dual.in	dual.out
7 0 1 2 2 2 0 2	5 1 6 2 0 3 0 4 5 7 0

Задача E. Elections (Division 1 Only!)

Имя входного файла:	`elections.in`
Имя выходного файла:	`elections.out`
Ограничение по времени:	8 seconds
Ограничение по памяти:	256 мебибайт

Во Флатландии парламентские выборы проходят по пропорциональной системе, то есть по партийным спискам. Каждая партия создаёт список своих кандидатов в парламент. После того, как население проголосует на выборах, места в парламенте делятся пропорционально количеству голосов, поданных за каждую партию. Выделенные каждой прошедшей в парламент партии места достаются кандидатам, расположенным в верхней части её списка.

Последние парламентские выборы, однако, стали причиной серьёзной политической дискуссии. Дело в том, что многие популярные общественные деятели были включены в избирательные списки различных партий, что увеличило процент поданных за данные партии голосов (и соответственно — количество мест в парламенте), однако после выборов эти деятели отказались от мест в парламенте в пользу кандидатов, находящихся ниже в партийном списке. Получилось, что избиратели обмануты: они голосовали за знаменитостей, которые не собирались работать в парламенте...

В качестве решения предлагалось отдавать оставленные знаменитостями места места представителям других партий, а не «следующим в списке». Однако в случае, когда представители нескольких партий покидают парламент, ситуация усложняется.

В результате перед следующими выборами была предложена новая схема формирования парламента. Пусть в выборах участвует n партий, которые борются за s мест в параменте. Перед выборами каждая партия формирует список кандидатов и происходит голосование за партийные списки. Пусть всего проголосовали V человек и за список i-й партии было подано v_i голосов.

Далее каждая партия составляет список делегатов, которые отказываются от своих мест, затем формируется «предварительный» список мест, в котором i-й партии предоставляется a_i предварительных мест. Пусть сумма всех a_i равна A. Пусть b_i — количество кандидатов, которые планируют работать в парламенте, среди первых a_i мест партийного списка для i-й партии, причём сумма $\sum\limits_{i=1}^{n} b_i$ должна быть равна s. Тогда соответствующие b_i представителей i-й партии получают места в парламенте.

Для того, чтобы распределение парламентских мест было максимально честным, числа a_i должны быть выбраны таким образом, чтобы значение суммы $\sum\limits_{i=1}^{n} \left|\frac{a_i}{A} - \frac{v_i}{V}\right|$ было минимальным.

Ваша задача — по результатам выборов и спискам кандидатов, отказавшихся от мест в парламенте, найти оптимальные значения a_i.

Формат входного файла

Первая строка входного файла содержит два целых числа n и s ($2 \leqslant n \leqslant 10$, $1 \leqslant s \leqslant 50$) — количество партий и количество мест в парламенте, соответственно.

В последующих $2n$ строках задаётся информация о каждой из партий. Информация о каждой партии задаётся в двух последовательных строках. Для i-й партии в первой из этих строк задано целое число v_i — количество голосов, поданное на выборах за эту партию ($0 \leqslant v_i \leqslant 10^4$, как минимум одно v_i строго положительно). Вторая строка начинается с целого числа m_i — количество кандидатов из списка i-й партии, которые собираются отказаться от мест в парламенте. Далее следуют m_i целых положительных чисел, упорядоченных по возрастанию и не превосходящих 10^4 — номера этих кандидатов в партийном списке.

Гарантируется, что сумма значений m_i для всех партий не превосходит 50.

Формат выходного файла

Выведите n целых чисел — оптимальные значения a_i.

Примеры

elections.in	elections.out
3 8 10 3 1 2 3 5 0 5 0	5 3 3

Задача F. Free of Squares

Имя входного файла:	`free.in`
Имя выходного файла:	`free.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 мебибайт

Рассмотрим слова над двоичным алфавитом $B = \{0, 1\}$. Возьмём число w длины l и занумеруем позиции символов в этом слове слева направо числами от 1 до l. Например, если $w =$ “0110”, то $w[1] = 0$, $w[2] = 1$, $w[3] = 1$ и $w[4] = 0$.

Позиция с номером i (где i пробегает значения от 1 до l) называется *позицией, свободной от квадратов*, если для все k таких, что $i + 2k - 1 \leqslant l$ слова $w[i \ldots i + k - 1]$ и $w[i + k \ldots i + 2k - 1]$ различны. Например, в слове “0110” позиция 1 свободна от квадратов, так как “0”≠“1” и “01”≠“10”, а позиция 2 нет, так как “1”=“1”.

Обозначим за $\sigma(k)$ количество позиций в слове w, имеющих номера от 1 до k включительно и свободных от квадратов. Например, для слова “0110” получим $\sigma(1) = 1$, $\sigma(2) = 1$, $\sigma(3) = 2$, $\sigma(4) = 3$.

Слово w длины l называется *довольно свободным от квадратов*, если для всех k от 1 до l включительно выполнено следующее неравенство: $2\sigma(k) \geqslant k$. К примеру, слово “0110” является довольно свободным от квадратов, а слово “0011” — нет.

По заданному l найдите все довольно свободные от квадратов слова длины l.

Формат входного файла

Во входном файле задано одно целое число l ($1 \leqslant l \leqslant 40$).

Формат выходного файла

Первая строка выходного файла должна содержать k — количество двоичных слов длины l, довольно свободных от квадратов. Последующие k строк должны содержать эти слова, отсортированные лексикографически, по одному слову на строку.

Примеры

free.in	free.out
4	4 0100 0110 1001 1011

Задача G. Gas Transportation

Имя входного файла:	`gas.in`
Имя выходного файла:	`gas.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 мебибайт

Флатландия планирует экспортировать газ в Угландию по уже построенной системе газопроводов, содержащей n газораспределительных станций, занумерованных целыми числами от 1 до n и m труб, при этом каждая труба соединяет две станции и может прокачивать миллион кубометров газа в месяц в выбранном направлении. При этом система построена так, что можно наладить транспортировку газа между двумя любыми станциями. Компания ФлатГаз — основной флатландский экспортёр газа — подаёт добываемый газ на станцию с номером 1. Газораспределительные сети Угландии подключены к станции с номером n.

Ситуация осложнчется тем, что у каждой из m труб имеется компания-собственник, которая требует оплаты за прокачку газа по трубе. Для того, чтобы уменьшить риски и оптимизировать расходы, ФлатГаз создал Газотранспотрную Ассоциацию (ГТА) и пригласил в неё всех собственников труб. Если владелец i-й трубы присоединяется к ассоциации, то он получает за прокачку газа по его трубе w_i флатландских долларов ежемесячно.

Однако компании-владельцы труб решили изучить ситуацию на рынке и сформировать альтернативное предложение. Зная рыночную цену c транспортировки одного миллиона кубометра газа из Флатландии в Угландию, они планируют найти такое подмножество труб A, по которому можно ежемесячно прокачать $f(A)$ миллионов кубометров газа от станции 1 до станции n. После чего собственники труб из этого множества создают альтернативный картель, который выставляет ФлатГазу счёт на $f(A) \cdot c$ флатландских долларов в месяц.

Назовём предложение ФлатГаза *честным*, если не существует такого множества труб A, по которому можно прокачать $f(A)$ миллионов кубометров газа за месяц и для которого значение $w(A) = \sum_{i \in A} w_i$ is строго меньше, чем $f(A) \cdot c$.

Вам задана схема трубопроводов, а также предложение ФлатГаза $\{w_i\}$. Проверьте, является ли оно честным. Если не является, найдите некоторое подмножество труб A, владельцы которых смогут сформировать альтернативный картель и получить больший доход, исходя из рыночной цены.

Формат входного файла

Первая строка входного файла содержит три целых числа: n, m и c ($2 \leqslant n \leqslant 500$, $1 \leqslant m \leqslant 5000$, $1 \leqslant c \leqslant 10^9$).

Последующие m строк задают трубы, при этом каждая труба задаётся в отдельной строке тремя целыми числами: a_i и b_i — станции, которые соединяет труба, w_i — предложение ФлатГаза владельцу этой трубы ($0 \leqslant w_i \leqslant 10^9$).

Формат выходного файла

Если предолжение Флатгаза является честным, выведите "`Fair`". Иначе выведите "`Unfair`", в следующей строке — количество элементов в множестве A и в третьей строке — номера труб, входящих в множество A. Трубы пронумерованы от 1 до m в порядке, в котором они заданы во входном файле.

Если существует несколько таких множеств, выведите любое из них.

Примеры

gas.in	gas.out
4 5 10 1 2 5 1 3 5 2 3 0 3 4 5 2 4 5	Fair
4 5 10 1 2 4 1 3 6 2 3 0 3 4 4 2 4 6	Unfair 3 1 3 4

Задача H. Handsome Division

Имя входного файла:	handsome.in
Имя выходного файла:	handsome.out
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 мебибайт

Простым числом называется число, которое делится нацело только на два целых числа: единицу и само себя. Последовательность простых чисел начинается так: $2, 3, 5, \ldots$.

Рассмотрим первые n простых чисел. Разобьём их на две части A и B так, что каждое простое число принадлежит одной и только одной из этих частей.

Обозначим произведение всех простых чисел из множества A как a, а произведение всех простых чисел из множества B как b. Разбиение называется *красивым*, если $a < b$ и разность $b-a$ минимальна.

По заданному n найдите красивое разбиение множества первых n простых чисел и выведите соответствующее значение a.

Формат входного файла

Входной файл содержит несколько тестовых примеров. Каждый тестовый пример состоит из одной строки, содержащей одно целое число n ($1 \leqslant n \leqslant 30$). За последним тестовым примером следует строка, в которой $n = 0$. Эту строку обрабатывать не требуется.

Формат выходного файла

Для каждого тестового примера выведите значение a для красивого разбиения множества первых n простых чисел. Формат вывода приведён в примере к задаче.

Примеры

handsome.in	handsome.out
1 2 3 4 5 0	Case #1: 1 Case #2: 2 Case #3: 5 Case #4: 14 Case #5: 42

Задача I. In Touch

Имя входного файла:	`intouch.in`
Имя выходного файла:	`intouch.out`
Ограничение по времени:	6 seconds
Ограничение по памяти:	256 мебибайт

Разработчик социальной сети "In Touch" Пол Умнов решил изменить принцип работы френдленты.

Пользователи сети могут оставлять сообщения на так называемой "стене", создавая, таким образом, свой личный микроблог. Некоторые пользователи являются друзьями, и для каждого пользователя существует *френдлента*, содержащая все сообщения от его друзей (но не от него самого).

До модификации пользователь мог видеть во френдленте все сообщения других пользователей, являющихся его друзьями. Согласно новым принципам работы френдленты пользователь видит только сообщения, которые были оставлены другими пользователями, являвшимися его друзьями на момент появления сообщения.

Когда два пользователя становятся друзьями, новые сообщения, написанные одним из них, показываются во френдленте другого. Если они перестают быть друзьями, сообщения, которые они писали, ещё будучи друзьями, остаются в их френдлентах, но новые их сообщения туда уже не попадают.

Для того, чтобы собрать некоторую статистику, Пол хочет узнать, из какого количества сообщений состоит френдлента каждого пользователя на текущий момент.

У каждого из n пользователей есть уникальный идентификационный номер — целое положительное число, не превосходящее n. Вам задан список событий в хронологическом порядке. События могуи быть трёх типов: "пользователь x написал сообщение в своём микроблоге "пользователи x и y стали друзьями" или "пользователи x и y перестали быть друзьями". При этом изначально никакие два пользователя друзьями не являются.

Выведите для каждого пользователя количество сообщений, которые окажутся в его френдленте после всех событий списка.

Формат входного файла

Первая строка входного файла содержит два целых числа: n — количество пользователей и m — количество событий в списке ($1 \leqslant n \leqslant 200\,000$, $0 \leqslant m \leqslant 500\,000$). Последующие m строк описывают события. Каждая из этих строк может содержать одно из следующих событий:

- "`!` x" — пользователь x написал сообщение в своём микроблоге;
- "`+` x y" — пользователи x и y стали друзьями;
- "`-` x y" — пользователи x и y перестали быть друзьями.

При этом $x \neq y$ — целые положительные числа, не превосходящие n.

Формат выходного файла

Выведите n целых чисел — количество сообщений во френдленте каждого пользователя. Количество сообщений во френдленте пользователя с идентификационным номером i должно идти i-м по порядку.

Примеры

intouch.in	intouch.out
4 10 ! 1 + 1 2 + 1 3 ! 1 ! 2 - 1 2 ! 1 ! 2 ! 3 ! 1	2 1 3 0

Задача J. Jumbo World

Имя входного файла:	`jumbo.in`
Имя выходного файла:	`jumbo.out`
Ограничение по времени:	5 seconds
Ограничение по памяти:	256 мебибайт

По примеру игры “Small World” компании “Days of Wonder” Петя решил сделать свою собственную игру и назвал её “Jumbo World”.

Игра происходит на доске, состоящей из нескольких зон. Некоторые зоны являются соседними, при этом у одной зоны может быть не более 5 соседей. Некоторые зоны являются соседними с границей доски. Эти зоны называются *граничными зонами*. Для каждой зоне определён тип её территории, от которого зависят различные бонусы для игроков.

В игре участвуют несколько человек, каждый из них управляет одной *расой*. Разные игроки контролируют разные расы. У каждой расы есть несколько *юнитов*, которые могут занимать различные зоны. Каждая зона может быть занята одним или несколькими юнитами одной расы. Юниты разных рас не могут занимать одну и ту же зону.

Игроки делают ходы по очереди. Ход состоит из «завоеваний» какого-то количества зон. Изначально игрок может завоевать любую из зон на границе. После того, как игрок завоевал зону на границе, он может продолжать завоевывать зоны, соседние с зонами, которые он уже завоевал.

У игрока есть несколько юнитов его расы. Завоевание зоны требует $\max(1, 2 + c + b)$ юнитов, где c — количество юнитов другой расы, занимающих территорию и b — значение бонуса, определяемое типом территории завоёвываемой зоны, свойствами расы, которая завоёвывает и расы, которая занимает эту территорию перед завоеванием. Юниты, участвующие в завоевании некоторой области, остаются в ней до конца хода.

Если у игрока не хватает юнитов для того, чтобы завоевать хотя бы какую-то из зон, соседних с уже завоёванными, его ход заканчивается и участнику начисляются очки за его зоны. Каждая зона по умолчанию стоит 1 очко, это количество может быть изменено бонусами. Также бонусы могут зависеть от свойств завоеваний.

Расы обладают различными свойствами, которые изменяют бонус за завоевание b и очки, получаемые в конце хода. Существуют следующие свойства:

Свойство	Описание
+k defence	Если зона, занимаемая этой расой, подвергается завоеванию, то требуется юнитов на k больше, чем обычно (b += k).
+k defence on T	(T — тип территории) Если зона, имеющая тип территории T и занимаемая данной расой, подвергается завоеванию, то завоевателю требуется k дополнительных юнитов. (b += k).
+k defence near T	(T — тип территории) Если среди соседей зоны X, занимаемой данной расой, имееется зона с типом территории T, то при завоевании зоны X требуется k дополнительных юнитов. (b += k).
+k defence versus R	(R — раса) Если территория, занимаемая данной расой, подвергается завоеванию со стороны расы R, то завоевателю требуется k дополнительных юнитов (b += k).
+k attack	Если данная раса захватывает некоторую зону, то для завоевания требуется на k юнитов меньше (b −= k).
+k attack on T	(T — тип территории) Если данная раса захватывает некоторую зону с типом территории T, то для завоевания требуется на k юнитов меньше (b −= k).
+k attack near T	(T — тип территории) Если данная раса захватывает некоторую зону, среди соседей которой есть зона с типом территории T, то для завоевания требуется на k юнитов меньше (b −= k).
+k attack versus R	(R — раса) Если данная раса захватывает некоторую зону, занимаемую расой R, то для завоевания требуется на k юнитов меньше (b −= k).
+k points	Данная раса получает за каждую занимаемую ей зону k дополнительных очков в конце хода.
+k points on T	(T — тип территории) Данная раса получает за каждую занимаемую ей зону, тип территории которой равен T, k дополнительных очков в конце хода.
+k points for conquest	Данная раса получает k дополнительных очков в конце хода за каждую завоёванную зону, в которой перед завоеванием находился как минимум один юнит чужой расы ($c > 0$).

При этом для всех свойств $1 \leqslant k \leqslant 9$.

Вы собираетесь сделать ход в игре “Jumbo World”, имея в распоряжении n юнитов заданной расы. На данный момент вы не занимаете ни одной зоны, однако некоторые зоны могут быть заняты юнитами других рас. Какое наибольшее количество очков вы можете получить в конце этого хода?

Формат входного файла

Первая строка входного файла содержит четыре цнлых числа t, r, a and n — количество типов территории, рас и зон в игре, а также количество имеющихся в вашем распоряжении юнитов, соответственно ($1 \leqslant t \leqslant 10$, $2 \leqslant r \leqslant 10$, $1 \leqslant a \leqslant 30$, $1 \leqslant n \leqslant 10$).

Каждая из последующих t строк содержит по одному слову, состоящему из строчных латинских букв — названия типов территории.

Далее следуют описания r рас. Каждое описание начинается со строки, содержащей название расы, затем следует строка, содержащая число v_i — количество свойств расы. Далее следуют v_i строк, каждая из которых содержит описание одного из свойств в соответствии с условиями задачи. Название каждой расы является одним словом, составленным из строчных латинских букв. Гарантируется, что среди свойств каждой расы не будет двух независимых от расы оппонентов и типа территории однотипных свойств, а также двух однотипных свойств, относящихся к одному и тому же типу территории или к одной и той же расе оппонентов.

Далее следует описание зон. Зоны занумерованы по порядку числами от 1 до a. Описание каждой

зоны содержит четыре последовательные строки. В первой из этих строк содержится тип территории для данной зоны. Вторая строка содержит слово "`border`", если данная зона является граничной, или "`interior`" в противном случае. Третья строка содержит название расы, которая занимает данную зону перед началом вашего хода, за которым задано одно целое неотрицательное число, не превосходящее 10 — количество юнитов, которые её занимают. Если зона не занята ни одной расой, третья строка содержит слово "`empty`". Гарантируется, что ни у одной расы нет названия "`empty`". Четвёртая строка начинается с b_i — количества зон, соседних с данной, за которой следуют b_i номеров этих зон. При этом ни одна зона не может быть соседней с самой собой. Если зона x является соседней с зоной y, то и зона y является соседней с зоной x. Гарантируется, что можно пройти из любой зоны в любую другую, каждый раз переходя только в соседние зоны и что как минимум одна зона является граничной.

Последняя строка входного файла содержит название расы, за которую вы играете.

Формат выходного файла

Выведите одно число — максимальное количество очков, которое вы сможете получить в конце этого хода.

Примеры

jumbo.in	jumbo.out
3 3 6 8 grass hills mountains dwarves 1 +2 points on mountains elves 3 +1 attack on grass +1 defence on grass +1 points for conquest human 2 +1 points on grass +1 attack versus dwarves mountains border dwarves 1 4 2 3 4 5 grass border elves 2 4 1 3 5 6 hills border elves 1 4 1 2 4 6 hills interior empty 4 1 3 5 6 hills interior empty 4 1 2 4 6 grass interior empty 4 2 3 4 5 human	5

Вы можете получить 5 очков, завоевав зону 1 с помощью $2 + 1 - 1 = 2$ юнитов и потом завоевав зоны 4, 5 и 6, потратив по 2 юнита на каждую из них.

Задача K. King and his network (Division 2 Only!)

Имя входного файла:	`king.in`
Имя выходного файла:	`king.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

Король Байтландии зарегистрировал аккаунт в местной социальной сети и пригласил нескольких придворных присоединиться. Некоторые из них пригласили новых участников, те, в свою очередь, новых и так далее. Сейчас в социальной сети зарегистрировано N человек с идентификационными номерами от 1 до N. При этом каждый пользователь, кроме Пола, приглашён ровно одним участником.

Король собирается «почистить» список приглашённых с помощью функции «удалить пользователя». Функция работает следующим образом: если она удаляет пользователя, то она удаляет и всех, кого он пригласил (а также кто был приглашён теми, кого он пригласил и так далее).

Считая, что король не может удалить самого себя, вычислите, сколько различных множеств пользователей он может удалить (вариант «не удалять никого» тоже является допустимым).

Формат входного файла

В первой строке входного файла задано целое число N ($2 \leqslant N \leqslant 6$) — количество участников социальной сети.

Далее следует $N - 1$ строка — информация о том, кто кого пригласил. В i-й из этих строк содержится идентификационный номер пользователя, пригласившего пользователя с номером i. Идентификационный номер N взял себе король.

Формат выходного файла

Выведите одно целое число — количество различных множеств участников, удалённых королём.

Примеры

king.in	king.out
3 3 3	4
4 3 4 4	6

Задача L. Lotto in the Ring (Division 2 Only!)

Имя входного файла:	`lotto.in`
Имя выходного файла:	`lotto.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

В игре «Лото в круге» требуется размещать монеты в нарисованном на координатной плоскости круге с центром в точке $(0, 0)$. Монета может быть размещена в каждой точке с целыми координатами, лежащей внутри круга или на окружности.

Ваша задача — по заданному радиусу круга вычислить, какое наибольшее количество монет можно разместить в этом круге.

Формат входного файла

Входной файл состоит из нескольких тестовых примеров. Каждый тестовый пример представляет из себя одно целое положительное число, не превосходящее $25 \cdot 10^3$ — радиус круга. За последним тестовым примером следует строка, в которой задано число 0. Эту строку обрабатывать не требуется. Гарантируется, что две монеты, размещённые в различных точках с целыми координатами, не будут соприкасаться.

Формат выходного файла

Для каждого тестового примера в отдельной строке выведите наибольшее количество монет, которое можно разместить в круге заданного радиуса.

Пример

lotto.in	lotto.out
2 3 4 0	13 29 49

Задача M. Map of City (Division 2 Only!)

Имя входного файла:	`map.in`
Имя выходного файла:	`map.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

Вам необходимо найти дорогу с одного конца города на другой. В городе есть много перекрёстков, и на каждом из этих перекрёстков существуют определённые ограничния на передвижение, вызванные постоянно идущим в городе ремонтом. Некоторые перекрёстки закрыты из-за ремонта полностью, по некоторым можно двигаться только в направлении с севера на юг и обратно, по некоторым, наоборот, в направлении с запада на восток и обратно, и по некоторым можно двигаться в любую из четырёх сторон света.

Ваши друзья, живущие в этом городе, снабдили Вас планом города, на котором отмечено состояние каждого перекрёстка. Каждый перекрёсток может быть обозначен одним из четырёх символов:

- Символ '+' обозначает, что перекрёсток открыт для движения как с запада на восток и обратно, так и с севера на юг и обратно.
- Символ '-' обозначает, что перекрёсток открыт только для движения с запада на восток и обратно.
- Символ '|' обозначает, что перекрёсток открыт только для движения с севера на юг и обратно.
- Символ '*' обозначает, что перекрёсток закрыт полностью.

Ваша задача — найти минимальное количество перекрёстков, которое Вы пройдёте, чтобы добраться от северо-западного угла города до его юго-восточного угла.

Формат входного файла

В первой строке входного файла задано целое число t ($1 \leqslant t \leqslant 10$) — количество тестовых примеров. Каждый тестовый пример начинается со строки, содержащей целое число r, далее следует строка, содержащая целое число c — размер плана города ($1 \leqslant r, c \leqslant 20$). Последующие r строк содержат c символов каждая. Каждый символ может быть или '+', или '*', или '-', или '|'. Самая северная сторона города задаётся в первой строке. Гарантируется, что перекрёсток на северо-западном углу города не отмечен звёздочкой.

Формат выходного файла

Для каждого тестового примера в отдельной строке выведите одно целое число — минимальное количество перекрёстков, которое Вы пройдёте, чтобы добраться от северо-западного угла города в юго-восточный. Если такого маршрута не существует, выведите -1.

Пример

map.in	map.out
3 2 2 -| *+ 3 5 +||*+ +++|+ **--+ 2 3 +*+ +*+	3 7 -1

Задача N. New Fan Club (Division 2 Only!)

Имя входного файла:	`newfans.in`
Имя выходного файла:	`newfans.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

Перед финалом TCO был образован новый фан-клуб Петра Митричева. Для начала клуб решил приготовить таблички с текстом "`WELCOME TO TCO GOOD LUCK PETR!`" и разместить их по пути следования Петра из аэропорта. Ширины таблички хватает, чтобы разместить в каждом ряду w символов, включая пробелы. При форматировании сообщения фанаты решили действовать следующим образом: во-первых, в текущей строке должно быть как можно больше слов (пока позволяет ширина); первое слово каждой строки начинается с самой левой позиции. Если в строке более одного слова, последний символ последнего слова находится в самой правой позиции; слова в строке разделены одним или несколькими пробелами; если требуется вставка дополнительных пробелов, то они вставляются в промежутки между словами так, чтобы промежутки были как можно более одинаковыми (то есть самый длинный и самый короткий отличались не более, чем на один пробел). Если все промежутки не могут быть сделаны одинаковыми, то большие промежутки должны идти перед меньшими.

По заданной ширине таблички изобразите табличку в соответствии с данными правилами.

Формат входного файла

В первой строке входного файла задано одно целое число w ($7 \leqslant w \leqslant 33$) — ширина таблички.

Формат выходного файла

Выведите получившуюся табличку. Вместо пробелов используете точку ('.').

Примеры

newfans.in	newfans.out
15	WELCOME..TO.TCO GOOD.LUCK.PETR!
26	WELCOME..TO..TCO.GOOD.LUCK PETR!.....