Задача A. Коллекционеры

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вчера Тая посещала музей. Экскурсия была долгой и интересной, но особенно ей понравилась комната, где были собраны коллекции кубиков 10 известных людей. Один из кубиков ей сильно приглянулся, но она забыла, кому он принадлежал. Зато она помнит как выглядели 3 видимые стороны кубика и кто какие кубики коллекционирует. По этой информации вам следует определить имена коллекционеров, в чьих коллекциях мог быть данный кубик.

У каждого кубика 6 граней. На каждой грани записано число от 1 до 6, на разных гранях разные числа. Числа могут изображаться точками, арабским числом или римским числом. Вдобавок каждая из граней покрашена в один из цветов — черный (Black), белый (White), зеленый (Green), желтый (Yellow), голубой (Skyblue), красный (Red), оранжевый (Orange) и фиолетовый (Purple).

Ниже представлен список имен коллекционеров и параметры кубиков, которым удовлетворяет вся его коллекция:

John	Все числа обозначены точками
David	Все числа обозначены не римскими числами
Peter	Кубик полностью белый
Robert	Грани кубика черные или белые
Mark	Нечетные числа на белом фоне, четные — на черном
Paul	Все простые числа арабские и все арабские числа простые
Patrick	Одноцветный цветной (не черный и не белый) кубик
Jack	Все римские числа на желтом фоне
Max	Все грани разноцветные
Alex	Числа одной формы записи на одном фоне, разного — на разном

Формат входных данных

Входной файл состоит их трех строк, описывающих видимые стороны кубика.

Первый символ i-й строки c_i ($c_i \in \{B, W, G, Y, S, R, O, P\}$) — цвет i-ой стороны (черный, белый, зеленый, желтый, голубой, красный, оранжевый и фиолетовый соответственно). Далее через пробел записана строка, обозначающая число на стороне в одном из трех форматов:

1. От 1 до 6 символов «`.`» (ASCII 46), означающих что число записано точками и равно количеству этих точек;
2. Арабское число от 1 до 6;
3. Римское число, записанное символами «`I`» (ASCII 73) и «`V`» (ASCII 86).

Гарантируется, что представленный кубик принадлежит хотя бы одному коллекционеру.

Формат выходных данных

Выходной файл должен содержать одну строку, содержащую имена коллекционеров, которым может принадлежать данный кубик. Имена должны быть записаны через пробел в любом порядке.

Все имена коллекционеров должны быть из следующего списка: `John`, `David`, `Peter`, `Robert`, `Mark`, `Paul`, `Patrick`, `Jack`, `Max`, `Alex`.

Примеры

input.txt	output.txt
W .. W ... W	John David Peter Robert Jack Alex
B 2 W 3 B 6	David Robert Mark Jack
G 1 G 2 G V	Patrick
G 2 G 3 Y	David Paul Jack Alex
W . B 2 W III	Robert Mark

Задача B. Идеальный подарок

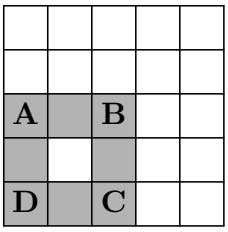
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Тая готовит подарок на день рождения. А самый лучший подарок — это подарок, сделанный своими руками. Недавно она научилось вышивать крестиком и решила воспользоваться этим умением для создания подарка.

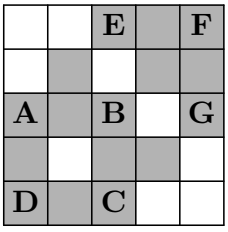
Дома у нее нашлась только канва, на которой ранее уже было вышито два крестика. Не беда — их можно дополнить до готовой картины. Опыта у нее мало, поэтому она выбрала довольно простой, но в то же время очень красивый рисунок, а именно параллелепипед. Ей хочется сделать подарок как можно скорее, поэтому количество дополнительных крестиков должно быть как можно **меньше**.

Параллелепипед на **бесконечном** клетчатом поле будем строить следующим образом.

Нарисуем прямоугольник $ABCD$ с верхним–левым углом A и нижним–правым C.



Проведем отрезки одинаковой длины вверх–вправо от вершин A, B и C — получим новые вершины E, F, G соответственно. Построим последние отрезки EF и FG.



Все длины сторон параллелепипеда должны быть **не менее 3-х клеток**.

Формат входных данных

В первой строке входного файла записано два целых числа x_1 и y_1 — координаты первого крестика. Во второй строке — координаты второго крестика: x_2, y_2. Координаты крестиков различны. Ось OX направлена слева направо, а ось OY направлена снизу вверх. Все числа принадлежат отрезку $[0, 10^9]$.

Формат выходных данных

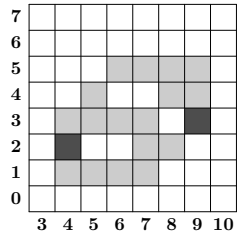
Выходной файл должен содержать одно число — **наименьшее** количество дополнительных крестиков.

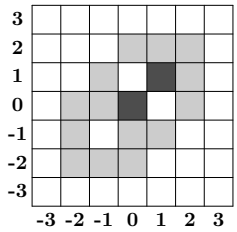
Примеры

input.txt	output.txt
4 2 9 3	17
0 0 1 1	14

Пояснение

Примерам соответствуют следующие рисунки:





Задача C. Казино

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Когда у Таи заканчиваются деньги, она идет играть в казино. Недавно в нем появилась новая игра, в которую Тая никак не может научиться играть оптимально. Помогите ей.

Игра проводится между крупье и посетителем казино. У крупье есть одна правильная k-гранная игральная кость, на гранях которой записаны все числа от 1 до k. В начале игры он берет кость и кидает один раз. Выпавшее число определяет количество очков крупье.

Для того чтобы выиграть, игроку требуется набрать больше очков, чем у крупье. Для этого ему предлагается выбор из n вариантов. Каждый вариант представляет собой пару: игральная кость и количество ее бросков. На каждой грани каждой кости написано какое-то число. Эта кость кидается нужное количество раз, все выпавшие числа суммируются и это значение и равно количеству очков игрока.

Но на некоторые грани, помимо обычных чисел, помечены как бонусные. Если выпала бонусная грань, то соотвествующее количество очков прибавляется к итогу, но за бросок это не считается. Все грани одной и той же кости попарно различны, то есть нет двух одинаковых бонусных граней и нет двух одинаковых обычных граней. На каждой кости есть по крайней мере одна грань, не являющаяся бонусной. Вероятность выпадения каждой грани одной игральной кости одинакова.

В задаче требуется для каждого возможного количества очков крупье от 1 до k определить номер варианта набора очков, при котором вероятность набрать строго больше очков, чем у крупье, максимальна.

Формат входных данных

Первая строка входного файла содержит одно целое число n ($2 \leqslant n \leqslant 10$) — количество игральных костей.

Следующие n строк содержат описания вариантов в следующем формате.

Первое число c_i ($1 \leqslant c_i \leqslant 10$) — количество бросков. Второе f_i ($2 \leqslant f_i \leqslant 12$) — количество граней. Следующие f_i строк v_{ij} — числа на гранях. v_{ij} является либо числом от 1 до f_i, обозначающим количество очков, либо числом от 1 до f_i с приписанным слева знаком плюс «+» (ASCII 43), обозначающим величину бонуса. В рамках одной кости все числа без знака плюс различны, все числа с приписанными плюсами различны, хотя бы одна грань не содержит знака плюс.

В последней строке записано одно целое число k, которое всегда равно $\max\limits_{1 \leqslant i \leqslant n} (c_i \times f_i)$.

Формат выходных данных

В выходном файле должно быть k строк, содержащих по одному целому числу b_i — номер наилучшего варианта, который позволит с наибольшей вероятностью набрать больше i очков (эта вероятность должна отличаться от правильного ответа не более чем 10^{-9}).

Кости нумеруются с 1 в том порядке, в котором перечислены во входном файле.

Примеры

input.txt	output.txt
3 3 4 1 2 3 4 2 6 1 2 3 4 5 6 1 12 1 2 3 4 5 6 7 8 9 10 11 12 12	2 1 1 1 1 1 1 3 3 3 3 2
2 1 4 1 2 +1 +2 1 6 1 +1 2 3 4 5 6	2 2 2 2 1 1

Пояснение

В ответе первого примера на первом месте может быть число 1, а последнее может быть любым от 1 до 3.

Задача D. В кубе

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Тая любит ходить с друзьями в кафе «В кубе», ведь там очень удобная система обслуживания. Чтобы сделать заказ, посетитель должен подойти к специальному стенду и выбрать понравившиеся блюда. Несколько таких стендов расположены на территории кафе в фиксированном положении.

В кафе посетители сидят за столами, которых k. За i-м столом может сидеть c_i человек. Неудобностью размещения стола назовем сумму расстояний от него до c_i ближайших стендов.

Формально, кафе представляет собой сетку $(0, 0) - (100, 100)$. В каждой клетке (x, y) $(0 \leqslant x, y \leqslant 100)$ может находиться один стенд, один стол, либо ничего.

Расстояние между клетками (x_1, y_1) и (x_2, y_2) равно $|x_2 - x_1| + |y_2 - y_1|$.

Требуется расставить столы так, чтобы сумма неудобностей всех столов была минимальной.

Формат входных данных

Первая строка входного файла содержит два целых числа n и k $(1 \leqslant n \leqslant 18,\ 1 \leqslant k \leqslant 200)$ — количество стендов и столов соответственно.

Следующие n строк содержат координаты i-го стенда: два целых числа x_i и y_i $(0 \leqslant x_i, y_i \leqslant 100)$.

Ось OX направлена слева направо, а ось OY направлена снизу вверх.

Далее k строк содержат по одному целому числу c_j $(1 \leqslant c_j \leqslant n)$ — количеству мест j-го стола.

Формат выходных данных

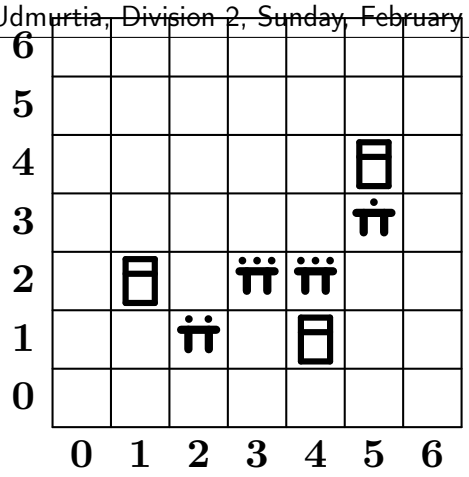
Выходной файл должен содержать одно целое число — минимальную сумму неудобностей.

Примеры

input.txt	output.txt
3 4 1 2 4 1 5 4 1 2 3 3	20
2 10 0 0 100 100 1 1 1 1 1 1 1 1 1 1	16

Пояснение

Возможный вариант расположения столов для первого примера выглядит так:



Задача E. Магазин игрушек

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Тая часто проходила мимо магазина игрушек и видела висящее около витрины электронное табло с двумя целыми числами. На витрине магазина лежат несколько видов игрушек, но числа на табло могут не соответствовать количеству видов игрушек. Оказалось, что не все игрушки с витрины можно купить, потому что их убирают с витрины не сразу, а спустя некоторое время после покупки. Для разных видов игрушек это время может быть разное.

Пусть есть n видов игрушек. Для каждого вида игрушки известно первоначальное количество c_i и количество минут t_i после покупки, через которое игрушка этого вида убирается с витрины. Каждую минуту происходит следующее:

- с витрины убирают игрушки, купленные соответствующее время назад;
- на табло обновляются числа;
- приходит новый покупатель и **обязательно покупает какую-то игрушку, оставшуюся в наличии**.

Таю всегда интересовал смысл чисел на табло и совсем недавно она его узнала. Оба числа показывали, сколько видов игрушек можно было приобрести в магазине, но первое показывало количество видов игрушек, **возможно** не распроданных к данному моменту времени, а второе — количество видов игрушек, **гарантированно** не распроданных к данному моменту времени. Также Тае интересно, насколько это табло информативно для посетителей. Поэтому ей нужна программа, которая будет моделировать поведение покупателей и пересчитывать табло.

Ваша задача — для каждой минуты определить числа на табло.

Формат входных данных

Первая строка входного файла содержит одно целое число n ($1 \leqslant n \leqslant 100$) — количество видов игрушек.

Следующие n строк содержат по два целых числа c_i и t_i ($1 \leqslant c_i \leqslant 100$, $1 \leqslant t_i \leqslant 100$) — количество игрушек i-го вида и время, через которое игрушку убирают с витрины после покупки соответственно.

Далее следует строка из одного целого числа k ($1 \leqslant k \leqslant 100$) — количества покупателей.

Следующие k строк содержат по одному целому числу q_i и q_i целых чисел $p_1, p_2, ..., p_{q_i}$ — количество игрушек, убранных в i-ую минуту и номера этих игрушек.

Формат выходных данных

Выходной файл должен содержать k строк, состоящих из двух целых чисел a_i и b_i — первое и второе число на табло в момент начала i-ой минуты соответственно.

Примеры

input.txt	output.txt
3 1 2 2 1 3 3 6 0 0 0 3 1 2 3 0 1 2	3 3 3 2 3 2 2 2 2 2 1 1

Пояснение

В приведенном выше примере на витрине находятся одна игрушка первого вида, две игрушки второго вида и три игрушки третьего вида, которые убирают после покупки через 2, 1 и 3 минуты соответственно. Числа на табло должны меняться в следующем порядке:

- 3/3: перед первым покупателем никого не было, он может купить любую игрушку.
- 3/2: первый покупатель мог купить игрушку первого вида, поэтому нет уверенности, что второй клиент может ее купить.
- 3/2: так как ни игрушку первого вида не убрали с витрины, ни второго, то это означает, что первый покупатель купил игрушку третьего вида. Что купил второй — пока определить нельзя.
- 2/2: игрушек первого вида больше не осталось.
- 2/2: никакую игрушку с витрины не убрали, значит предыдущий покупатель купил игрушку третьего вида.
- 1/1: Некупленной осталась только одна игрушка третьего вида.

Задача F. Игра с кубиками

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Тая нашла на чердаке очень старую игру, в которую она выигрывала через раз. Покажите ей, как можно выигрывать в нее гарантированно.

Для игры небходима круглая фишка радиуса 1, у которой сверху нарисована стрелка. Перед началом игры необходимо на столе пометить некую точку и положить фишку на стол.

Цель игры — накрыть точку фишкой. Этого нужно достичь, делая ходы по следующему правилу. В начале хода фишка поворачивается на целое число градусов от 0 до 359 против часовой стрелки и затем перекладывается по направлению стрелки на расстояние 10.

Помеченная точка всегда имеет координаты $(0, 0)$.

Найдите любую последовательность из не более 7500 ходов, приводящую к выигрышу.

Формат входных данных

Первая строка содержит два целых числа x, y ($2 \leqslant \max(|x|, |y|) \leqslant 500$) — начальные координаты центра фишки.

Следующая строка содержит два вещественных числа vx, vy, записанных с 9 знаками после запятой ($vx^2 + vy^2 = 100$) — направление стрелки.

Формат выходных данных

Первая строка должна содержать натуральное число k — количество ходов в найденной выигрышной последовательности ($1 \leqslant k \leqslant 7500$). Во второй строке через пробел должны записаны k целых чисел — значения углов поворота фишки на соответствующем ходе.

Примеры

input.txt	output.txt
10 -10 0.000000000 -10.000000000	2 180 90

Задача G. Data Mining

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Одна крупная IT-компания использует для анализа данных программу, которая была написана без распараллеливания. Когда требуется запустить эту программу, сотрудник запускает её и ждёт результата выполнения, тратя своё рабочее время.

Руководство компании рассматривает план, заключающийся в переписывании программы с использованием параллельных технологий. Чтобы проверить, окупится ли решение за год, используется следующая схема.

Сначала выясняется количество запусков программы в год, затем выясняется количество человеко-часов, которые уйдут на переписывание, затем оценивается время работы для параллельной версии программы, после чего проверяется, будет ли достигнута экономия в человеко-часах.

Выясните, будет ли эффективным переписывание программы в течение ближайшего года (то есть будет ли суммарное количество человеко-часов, затраченное на переписывание программы и на суммарное ожидание результата выполнения параллельной версии меньше количества человеко-часов, затраченных на суммарное ожидание результата выполнения существующей версии).

Формат входных данных

В первой строке входного файла задано целое число T — количество тестовых примеров ($1 \leqslant T \leqslant 1000$).

Каждый тестовый пример задан в одной строке и содержит четыре целых числа d, n, p и s — количество человеко-часов на разработку, количество запусков программы в год, время ожидания завершения работы параллельной версии, время ожидания завершения работы существующей версии соответственно. Время задаётся в часах, $0 \leqslant d \leqslant 10^6$, $0 \leqslant n \leqslant 10^5$, $0 \leqslant p, s \leqslant 1000$.

Формат выходных данных

Для каждого тестового примера выведите “`Rewrite`”, если переписывание программы даст выигрыш в человеко-часах, “`Keep`”, если оно даст проигрыш и “`Flip a Coin`”, если количество затраченных человеко-часов не зависит от решения о переписывании.

Пример

input.txt	output.txt
3 11 3 3 2 19 6 9 3 0 5 100 100	Keep Rewrite Flip a Coin

Задача H. Performance

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Производительность автомобильного двигателя с несколькими передачами для заданной передачи не является постоянной: она зависит от количества оборотов двигателя в минуту. Обычно эта зависимость задаётся так называемой кривой производительности.

Для серии двигателей “Параболика” кривая производительности представляет собой параболу $P = -aR^2 + bR + c$, где R — количество оборотов двигателя в секунду, а P — соответствующая производительность.

По заданным параметрам парабол, описывающих все передачи в двигателе, определите, на какой передаче достигается максимум производительности.

Гарантируется, что такая передача ровно одна и что значение максимальной производительности для этой параболы отличается от значений максимальной производительности для других парабол не менее, чем на 1.

Формат входных данных

В первой строке входного файла задано целое число T ($1 \leqslant T \leqslant 100$) — количество тестовых примеров.

Каждый тестовый пример начинается строкой, содержащей целое число n ($1 \leqslant n \leqslant 10$) — количество передач в двигателе. Далее следуют n строк, каждая из которых содержит по три целых числа a, b и c ($1 \leqslant a, b, c \leqslant 10^4$) — параметры соответствующей параболы. Передачи занумерованы с 1 в порядке, в котором они перечислены во входном файле.

Формат выходных данных

Для каждого тестового примера выведите одно целое число — номер передачи, позволяющей достичь максимума производительности.

Примеры

input.txt	output.txt
1 2 3 130 1423 2 153 210	2

Задача I. Tic-tac-toe

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Дана недоигранная партия в крестики-нолики на доске 3×3. Всего сделано 7 ходов. Вычислите, у кого из игроков выше вероятность выигрыша, если игрок, делающий следующий ход, сделает его случайным образом.

Формат входных данных

Входной файл состоит из 3 строк, задающих игровое поле. Каждая строка состоит из трёх символов; допустимыми символами являются (‘`o`’, ‘`x`’ или ‘`.`’, обозначающий свободную клетку). На доске осталось ровно две свободные клетки; гарантируется, что партия корректна (то есть разность между количеством крестиков и ноликов по модулю равна единице и не существует тройки одинаковых символов по горизонтали, вертикали и диагонали).

Формат выходных данных

Выведите ‘`o`’, если играющий ноликами имеет лучшие шансы на победу, ‘`x`’, если лучшие шансы на победу у крестиков, и “`tie`”, если шансы равны.

Примеры

input.txt	output.txt
xo. oxo xx.	x

Задача J. Move the Boxes

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

There are $m + 1$ positions located along a straight line, numbered by integers from 0 to m inclusive. The task is to move all the boxes from one position to another. You are allowed to use n cranes. Initially, all the cranes are in position 0. Each second, each crane can either stay on its position or move to an adjacent position to the left or to the right. Each crane can either be empty or hold a single box. The time required to attach the box to the crane or to detach it is negligible.

Find the least possible time required to move all k boxes from position 0 to position m. Each position can contain any number of boxes. Neither cranes nor boxes interfere with each other when moving.

Формат входных данных

The only line of input contains three integers n, m and k ($1 \leqslant n, m, k \leqslant 10^5$).

Формат выходных данных

Output a single integer: the number of seconds required to move all k boxes from position 0 to position m.

Примеры

input.txt	output.txt
1 10 5	90
5 10 5	10