Задача A. ACM contest

Имя входного файла:	`acm.in`
Имя выходного файла:	`acm.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	64 Mebibytes

При регистрации команд на 1/256 финала Чемпионата Срединной Империи по программированию выяснилось, что таблицы команд заданы в формате, отличающемся от Императорского стандарта. Требуется написать программу, переводящую список команд в Императорский стандарт.

Формат входного файла

Каждая команда в списке задаётся одной строкой. Данные в строке перечислены через точку с запятой (‘;’) в следующем порядке: сначала идёт рейтинг команды (целое положительное число, не превосходящее 1000), затем — её название (строка длиной от 1 до 25 символов, состоящая из цифр, строчных и заглавных латинских букв и пробелов, начинающаяся с цифры или буквы и заканчивающаяся цифрой или буквой). Следующие три поля задают список участников в формате «имя участника», символ slash (’/’), «факультет, на котором он учится». Ограничения на имя участника и название факультета такие же, как и для названия команды. Список завершается строкой ‘;STOP;’, которую обрабатывать не надо. Всего в списке содержится не более 5000 команд.

Формат выходного файла

Для каждой команды в порядке, заданном в исходном файле, выведите информацию о ней в Императорском стандарте:

- В первой строке выводится слово ‘Rating’, затем через пробел — рейтинг команды, затем двоеточие, затем пробел, затем название команды.
- В каждой из последующих трёх строк заданы участники в том порядке, в котором они были перечислены в исходной заявке. После имени каждого участника выводится запятая, затем выводится название факультета.
- В последней, пятой строке описания команды содержатся 10 знаков ‘-’.

Пример

acm.in
999;A Crush;Mit Ri Chew/Topcoding;Qu Lee Kou/GCJ;Mao Rin/ACM 990;To U Rist;Gen Na/School Go;Dy Kor Ot/School Mel Be;Ke Wich/School La Rus ;STOP;
acm.out
Rating 999: A Crush Mit Ri Chew, Topcoding Qu Lee Kou, GCJ Mao Rin, ACM ---------- Rating 990: To U Rist Gen Na, School Go Dy Kor Ot, School Mel Be Ke Wich, School La Rus ----------

Задача B. Boxes with treasure

Имя входного файла:	`boxes.in`
Имя выходного файла:	`boxes.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	64 Mebibytes

Для того, чтобы рыцари охотнее вступали с драконами в поединки, драконы держат в пещерах сокровища. Сокровища обычно спрятаны в сундуках так, что даже очень ловкий и сильный рыцарь должен затратить некоторое время на то, чтобы добраться до сокровищ. Один из вариантов — вложенные друг в друга последовательно сундуки. Сундуки представляют собой ящики в виде прямоугольного параллелепипеда с целой длиной стороны. Для каждого сундука заданы его длина x_l, ширина x_w и высота x_h. Сундук a может быть вложен в сундук b в том и только том случае, если a_l строго меньше b_l, a_w строго меньше b_w и a_h строго меньше b_h (то есть положение сундука в пространстве фиксировано и поворачивать сундук нельзя). В один сундук может быть непосредственно вложено не более одного сундука.

У дракона имеется N сундуков, для каждого из которых известна длина ребра. Требуется собрать эти сундуки описанным выше образом так, чтобы количество получившихся комплектов было минимально.

Формат входного файла

В первой строке входного файла задано одно число N — общее количество сундуков ($1 \leqslant N \leqslant 500$). Далее заданы N троек целых положительных чисел, не превосходящих 10^4, по три числа в каждой строке — соответственно длина, ширина и высота каждого из ящиков.

Формат выходного файла

Выведите одно число — минимальное количество комплектов, которые получатся, если собрать все сундуки описанным выше образом.

Примеры

boxes.in	boxes.out
3 6 5 9 28 11 11 101 33 524	1
3 2 3 2 3 2 2 2 2 3	3

Задача C. Colors

Имя входного файла:	`colors.in`
Имя выходного файла:	`colors.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	64 Mebibytes

На складе имеется несколько ёмкостей с красками. Каждая краска определяется своей RGB-палитрой — количество в литрах красной, зелёной и синей краски, из которых эта краска смешана. Можно ли, используя имеющиеся на складе ёмкости, получить в резервуаре заданное количество литров серой краски (то есть краски, в которой количество красной, зелёной и синей компонент будет одинаково)? При этом переливать краску из каждой ёмкости в резервуар можно только один раз, то есть частичное выливание или использование ёмкостей в качестве мерного инструмента не допускается.

Формат входного файла

В первой строке входного файла задано целое число N, делящееся на 3 — количество литров серой краски, которое нужно получить ($1 \leqslant n \leqslant 300$). В следующей строке задано целое число k ($1 \leqslant k \leqslant 100$) — количество ёмкостей с краской, имеющихся на складе. В последующих k строках перечислены сами ёмкости. Каждая ёмкость задаётся строкой с тремя целыми неотрицательными числами, не превосходящими 100 — количеством красной, зелёной и синей краски соответственно, из которых смешана краска в этой ёмкости.

Формат выходного файла

Выведите `'YES'`, если заданное количество краски можно получить, используя имеющиеся ёмкости, и `'NO'` в противном случае.

Пример

colors.in	colors.out
12 4 2 1 3 2 3 1 7 7 7 8 8 8	YES
24 2 3 1 2 1 3 2	NO
30 5 2 1 3 4 5 3 2 2 2 6 5 5 1 2 2	NO

Задача D. Domino

Имя входного файла:	`domino.in`
Имя выходного файла:	`domino.out`
Ограничение по времени:	40 seconds
Ограничение по памяти:	64 Mebibytes

Фишка «Треугольного домино» представляет собой два одинаковых правильных треугольника, имеющих общую сторону. В каждом из треугольников, подобно обычному домино, записаны цифры от 1 до 6. Пристыковывать две фишки треугольного домино можно по стороне, в случае, если пристыковываемая фишка не перекрывает уже выложенные, и если в треугольниках, расположенных по обе стороны от линии стыка, записаны одинаковые цифры. Если фишка пристыковывается к нескольким фишкам (разными сторонами), то соответствующее правило должно выполняться для каждого стыка; например, фишки (1,2), (2,3), (2,3) и (3,5) могут образовать «четырёхугольник»: к (1,2) пристыковываются по двум свободным сторонам треугольника с цифрой 2 фишки (2,3), после чего получается «место» для фишки (3,5):

```
    <2 3>
<1 2>   <3 5>
    <2 3>
```

Назовём «цепочкой» фигуру, которую можно получить, последовательно пристыковывая по одной фишке к уже выложенным (первая фишка просто выкладывается на поле). За каждые две фишки в «цепочке», соприкасающиеся сторонами, начисляется 1 балл. Так, цепочка на иллюстрации даёт 4 балла.

По заданному набору фишек треугольного домино выясните, какое максимальное количество баллов можно набрать, соединив фишки из этого набора в одну цепочку.

Формат входного файла

В первой строке входного файла задано одно целое число N ($1 \leqslant N \leqslant 6$) — количество фишек в наборе. В последующих N строках заданы сами фишки парами целых чисел от 1 до 6, записанных в треугольниках, составляющих данную фишку.

Формат выходного файла

Одно число — максимальное количество баллов, которое можно набрать, выложив фишки из данного набора в одну цепочку.

Пример

domino.in	domino.out
4 2 1 2 3 2 3 3 5	4
4 5 4 5 6 1 3 1 2	1
3 1 6 2 3 4 5	0

Задача E. Moon station

Имя входного файла:	`moon.in`
Имя выходного файла:	`moon.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	64 Mebibytes

Автоматический грузовой корабль совершил посадку на поверхности Луны. На борту аппарата имеется некоторое количество луноходов, предназначенных для доставки мини-лабораторий.

Программное обеспечение луноходов устроено так, что они могут двигаться только параллельно осям координат и поворачивать только в точках с целыми координатами (то есть можно считать, что они перемещаются по единичным квадратам клетчатой бумаги). Луноход перемещается по освещённой поверхности, не используя аккумуляторы. В случае, если луноход перемещается в тени, тратится 1/3 всего запаса энергии в аккумуляторе на 1 квадрат. Луноход с разряженным аккумулятором может двигаться только по освещённой поверхности.

Кроме того, на поверхности Луны имеются источники энергии, установленные предыдущей экспедицией. Используя эти источники, луноход может полностью зарядить батареи. Добравшись до точки назначения, луноход остаётся на месте в качестве конструктивного элемента мини-лаборатории и его возвращение назад на корабль не требуется.

Вам задана карта Луны, на которой отмечено место посадки корабля, освещённые и затенённые участки, а также расположение источников энергии и «целевые» квадраты, в которые должны быть доставлены мини-лаборатории (количество луноходов на борту корабля не меньше, чем количество «целевых» квадратов). Заметим, что луноход может проходить через «целевые» квадраты, если лаборатория на его борту предназначена для другого «целевого» квадрата.

Требуется выяснить, в какое количство «целевых» квадратов удастся добраться луноходам с мини-лабораториями.

Формат входного файла

В первой строке входного файла заданы два целых числа N и M, $(1 \leqslant N, M \leqslant 50)$ — размер карты в квадратах. Далее следуют M строк по N символов в каждой. Символ `'S'` обозначает место посадки корабля, символ `'.'` — освещённую поверхность, символ `'D'` — затенённую поверхность, символ `'G'` — «целевой» квадрат, символ `'R'` — источники энергии, от которых луноход может подзаряжать аккумуляторы. При этом и целевые квадраты, и источники энергии освещены.

Формат выходного файла

Выведите одно число — количество «целевых» квадратов, в которые могут добраться луноходы с мини-лабораториями.

Пример

moon.in	moon.out
20 8 S.......DDDD..GGG... DDD.....DDDDR.GGG... GGD.GG...DDDDDDDDDDD ..D.......DDDDDDDDDD G.D.......DDDDDDDDDD DDDDDDDDDDDDDDDDDDDD ..DDDDDDDDDRDDDRDDD G.DDDD.G.RDDDRDDDDDG	8

Задача F. Script

Имя входного файла:	`script.in`
Имя выходного файла:	`script.out`
Ограничение по времени:	16 seconds
Ограничение по памяти:	64 Mebibytes

В текстовом редакторе «wj» скрипт для редактирования состоит из следующих команд:

- `'a'` — вывести один символ. Аргумент — выводимый символ. Указатель чтения не сдвигается. Команда выполняется 1 такт.
- `'c'` — копировать один символ. Команда копирует символ из входной строки. Указатель чтения сдвигается на 1 символ вперёд. Аргумент — копируемый символ (используется только для контроля). Команда выполняется мгновенно.
- `'d'` — удалить один символ. Команда считывает один символ из входной строки. Указатель чтения сдвигается на 1 символ вперёд. Аргумент — удаляемый символ (указывается только для контроля). Команда выполняется 1 такт.
- `'m'` — изменить один символ. Команда считывает один символ из входной строки и выводит другой. Указатель чтения сдвигается на 1 символ вперёд. Аргумент — символ, который надо вывести. Команда выполняется 1 такт.

В начале работы редактора указатель стоит на начале исходной строки. Требуется написать программу, которая за минимальное число тактов переводит одну строку в другую. В начале работы редактора указатель стоит на начале исходной строки.

Формат входного файла

В первой строке входного файла задана исходная строка, во второй — строка, которую нужно получить. Строки могут состоять из заглавных и строчных латинских букв, а также цифр. Обе строки непусты, длина каждой строки не превосходит 18000.

Формат выходного файла

Выведите программу для редактора, которая бы за минимальное число тактов преобразовывала первую строку во вторую. Каждая команда записывается в отдельной строке и должна состоять из двух символов (первый символ — команда, второй — аргумент), разделённых ровно одним пробелом. При этом строгое соблюдение формата вывода (отсутствие лишних пробелов в конце вывода и вывод ровно одного символа перевода строки в конце каждой строки) обязательно.

Пример

script.in	script.out
1234E a12q4Ez	a a c 1 c 2 m q c 4 c E a z