Задача A. Метеориты

Игровое поле в игре «Метеориты» представляет собой полоску длиной N клеток и высотой в 4 клетки. В некоторых клетках поля расположены метеориты, при столкновении с которыми космический корабль, управляемый игроком, разбивается.

Для управления кораблём используется три команды: "U" — перемещает корабль на одну клетку вверх, "D" — перемещает корабль на одну клетку вниз и "R X" — перемещает корабль на X клеток вправо (независимо от числа X это считается одной командой).

Клетки игрового поля занумерованы с верхнего левого угла, верхняя левая клетка имеет X координату 1 и Y координату 1. В начале игры корабль также находится в верхней левой клетке поля.

Среди игроков считается наиболее престижным достичь столбца с номером N используя наименьшее количество команд.

Вам удалось узнать карты, на которых будет проводится чемпионат по игре в «Метеориты» и вы хотите пройти все карты, используя как можно меньше команд для управления кораблем. Гарантируется, что на всех картах можно добраться до столбца N, используя только разрешённые команды.

В первой строке входного файла задается число N — количество карт. Каждое описание карты состоит из чисел N и K — длины полоски и количества метеоритов на ней. В следующих K строках задаются координаты метеоритов в виде двух чисел X и Y. Метеориты перечисляются в произвольном порядке. Карты разделяются одной пустой строкой.

В качестве ответа необходимо вывести N описаний последовательности команд. Каждое описание последовательности команд должно состоять из числа C — количества команд и C команд по одной в строке.

Каждая последовательность команд оценивается отдельно, баллы за последовательность можно получить только в случае, если космический корабль не врезался в метеорит и достиг любой клетки столбца N. Балл вычисляется по формуле $(\frac{BestAns}{PartAns})^3$, где BestAns — наименьшее количество команд среди участников и жюри, а PartAns — количество команд у участника, сдавшего оцениваемое решение.

В первом тесте $N = 3$. Оценка за этот тест: 30 баллов. Проверка осуществляется в режиме on-line (результат виден сразу).

Во втором тесте $N = 7$. Оценка за этот тест: 70 баллов. Во время тура проверяется только соответствие формату данных (описано 7 последовательностей команд, в каждой последовательности количество команд соответствует указанному и все команды имеют вид "U", "D" или "R X"). Проверка правильности ответа осуществляется в режиме off-line (результат виден после окончания тура).

Примеры

Входные данные	Результат
2 10 0 10 4 2 1 4 4 2 2 2 3	1 R 9 6 D D D R 2 U R 7

Задача B. Экскаватор

Вася живёт в Вертикальной Дискретной Флатландии — двумерном мире, в котором существует только длина и высота, а весь мир разделен на квадраты с длиной стороны 1.

Вася очень любит горы, но его окрестности абсолютно плоские (для любой X-координаты высота в этой клетке равна 0). Поэтому он заказал экскаватор, который умеет выполнять единственный типа операций: насыпать на все клетки с позиции L до позиции R включительно слой земли высотой H.

Вася разработал несколько проектов ландшафта, который он хочет создать на клетках с координатами от 1 до K. Помогите Васе разработать как можно более короткую последовательность команд для экскаватора, которая позволит создать желаемый ландшафт.

В первой строке входного файла задается число N — количество проектов ландшафта, разработанных Васей. Каждое описание блока состоит из числа K — координаты до которой идёт строительство и K целых неотрицательных чисел — высоты столба земли на клетках от 1 до K.

В качестве ответа необходимо вывести N описаний последовательности команд. Каждое описание последовательности команд должно состоять из числа C — количества команд и C троёк чисел L, R, H, описывающих команду для экскаватора.

Решения проверяются, если все команды корректны (т.е. $1 \le L \le R \le K$). Каждая последовательность команд оценивается отдельно, баллы за последовательность можно получить только в случае, если сформирован правильный ландшафт и использовано менее K команд (гарантируется, что это всегда возможно). Балл вычисляется по формуле $(\frac{BestAns}{PartAns})^3$, где BestAns — наименьшее количество команд для формирования ландшафта среди участников и жюри, а PartAns — количество команд для формирования ландшафта у участника, сдавшего оцениваемое решение.

В первом тесте $N = 3$. Оценка за этот тест: 30 баллов. Проверка осуществляется в режиме on-line (результат виден сразу).

Во втором тесте $N = 7$. Оценка за этот тест: 70 баллов. Во время тура проверяется, что сданный файл содержит описание 7 корректных наборов команд, но не проверяется правильность построенного ландшафта. Проверка правильности ответа осуществляется в режиме off-line (результат виден после окончания тура).

Примеры

Входные данные	Результат
2 4 5 5 5 5 5 1 2 3 2 1	1 1 4 5 3 1 5 1 2 4 1 3 3 1

Задача C. Красивые деревья

Не так давно Гриша делал очередную задачку на олимпиаду. В этот раз его задачей было сгенерировать деревья с некоторыми нечётными приоритетами в каждой вершине. Гриша прочитал в интернете, что лучшим тестом для его задачи будет красивое дерево.

Красивым деревом с N узлами является такое дерево, в котором в каждой вершине можно расставить различные нечётные числа от 1 до $2 \cdot N - 1$ так, чтобы модули разности чисел, записанных в вершинах на концах каждого ребра, также были различны.

Также Гриша прочитал, что пока что не найдено дерева, которое было бы признано некрасивым, но и не существует доказательства того, что каждое дерево является красивым.

Тем не менее, олимпиада уже через неделю, а Грише позарез нужны тесты. Гриша решил, что ему необязательно расставлять приоритеты так, чтобы деревья были красивыми. Поэтому он сделает тесты так, чтобы количество рёбер с различными модулями разности было максимальным.

Вам предлагается решить аналогичную задачу. В каждом тесте из набора дано какое-то дерево. В первой строке записано число N — количество вершин в дереве и далее в $N - 1$ строке двумя числами a_i и b_i описаны рёбра. Эти числа значат, что из вершины a_i в веришну b_i существует ребро. Вам нужно расставить нечётные числа в вершинах и сдать файл с ответом в проверяющую систему. (см. пример в тесте из условия)

Каждое дерево оценивается независимо. Балл за дерево можно получить только в том случае, если каждой вершине было присвоено уникальное нечетное число. Балл вычисляется по формуле $\frac{PartAns \cdot (PartAns+1)}{BestAns \cdot (BestAns+1)} \cdot S$, где $PartAns$ — количество различных разностей приоритетов в дереве, найденных участником, а $BestAns$ — максимальное количество различных модулей разностей приоритетов в дереве, найденных жюри или участниками олимпиады, а S — максимальное количество баллов за тест.

Каждый набор тестов представлен папкой с тестами, пронумерованными по порядку. В файле с ответом в i-й строке должен быть ответ на i-й тест из набора тестов.

В первом наборе тестов дано 5 деревьев, каждое оценивается из $S = 6$ баллов (Итого 30 баллов). Проверка осуществляется в режиме on-line (результат виден сразу).

Во втором наборе тестов дано 14 деревьев, каждое оценивается из $S = 5$ баллов (Итого 70 балллов). Во время тура проверяется, что в каждом дереве использованы все нечётные числа от 1 до $2 \cdot N - 1$. Проверка осуществляется в режиме off-line (результат виден после окончания тура).

Примеры

Входные данные	Результат
5 1 2 3 2 2 4 4 5 4 1 2 2 3 2 4	1 9 3 5 7 3 7 1 5

Задача D. Сортировочная станция

Недавно король Байтландии издал приказ о создании железной дороги в стране. Так как раньше железной дороги в Байтландии не было, а использовать импортные технологии и наработки король не хочет, пришлось разрабатывать почти всё заново. Так случилось и с сортировочной станцией. НИИ ЧАГО разработало стандарт, по которому сортировочная станция работает следующим образом: машинисту посылается последовательность команд. Каждая команда состоит из четырёх слов. Первое слово — это L или R, с какой стороны надо подъехать к поезду. Машинист всегда может подъехать к поезду с той или другой стороны. Далее идёт число — сколько вагонов надо отцепить. Дальше ещё два числа — с какого и на какой путь перегнать отцепленные вагоны (сторону подъезда к поезду менять нельзя). То есть, если машинисту пришла команда "R 5 3 2", то он подъедет к поезду с правой стороны, отцепит пять вагонов с третьего пути и прицепит их так же справа к составу на втором. Если машинисту приходит некорректная команда (неположительное количество вагонов/большее, чем есть на путях/несуществующий путь), то он бросает работу и идёт домой.

В связи с этим самая большая проблема возникла на Битгородской сортировочной станции. С этой станции вагоны будут отправляться в один из двадцати шести пунктов назначения (их названия задаются буквами от A до Z). Чтобы было удобнее отправлять поезда, необходимо предварительно перегнать все вагоны одного направления на один путь. При том, на каком именно пути должен стоять тот или иной состав, в ГОСТе не оговорено, поэтому состав из вагонов, направляющуюся в одну точку назначения можно оставить на любом пути.

Сейчас идёт стадия планировки и прорабатывания различных ситуаций. Было решено нанять программиста, который сможет разработать наиболее оптимальные последовательности команд для машиниста, то есть минимизирует их количество.

Ситуация описывается следующим образом: в первой строке написано одно число N — количество путей. Далее в каждой из N описан состав, который стоит на этом пути. Каждый вагон задаётся буквой от A до Z — идентификатором пункта назначения. Количество различных пунктов назначения не больше, чем количество путей на станции. Если состава на пути нет, строка остаётся пустой.

В выходной файл для каждой ситуации строго по порядку запишите сначала одно число - количество команд, предписуемых к исполнению, а затем все эти команды. Если в процессе перестановок задача не выполнена, за ситуацию ставится 0 баллов.

Каждая ситуация оценивается отдельно. Баллы за каждую решённую ситуацию можно получить только в случае, если в процессе машинист не ушёл домой и все вагоны после манёвров отсортированы по месту назначения. Каждая ситуация оценивается из 10-ти баллов. Ситуации разделены пустой строкой. Итоговый балл вычисляется по формуле $\frac{BestAns \cdot (BestAns+1)}{PartAns \cdot (PartAns+1)} \cdot 10$, где $PartAns$ — количество команд, найденных участником а $BestAns$ — минимальное количество команд, найденных жюри или участниками олимпиады.

В первом файле описано 3 ситуации. Оценка за каждую ситуацию — 10 баллов (Итого: 30 баллов). Проверка осуществляется сразу в режиме on-line (результат виден сразу)

Во втором файле описано 7 ситуаций. Оценка за каждую ситуацию — 10 баллов (Итого: 70 баллов). Во время тура проверяется соответствие формата данных. Проверка осуществляется в режиме off-line (результат виден после тура)

Примеры

Входные данные	Результат
3 AAB BA 3 CC BC CA	3 L 2 1 2 L 1 3 1 R 1 3 2 2 R 1 2 1 L 1 3 1

Задача E. Беллман-Форд наоборот

Алгоритм Беллмана-Форда – алгоритм поиска кратчайшего пути во взвешенном графе, работающий за $O(|V| \times |E|)$ (где V — множество вершин графа, а E — множество его рёбер). В отличие от алгоритма Дейкстры алгоритм Беллмана-Форда допускает рёбра отрицательного веса.

Сформулируем задачу, решаемую алгоритмом Беллмана-Форда так: дан ориентированный граф G со взвешенными рёбрами. Длиной пути назовём сумму весов рёбер, входящих в этот путь. Требуется найти кратчайшие пути от выделенной вершины номер 1 до всех остальных вершин графа.

Построим матрицу A_{ij}, элементы которой будут обозначать следующее: A_{ij} — это длина кратчайшего пути из первой вершины в i, содержащего j рёбер.

Путь, содержащий 0 рёбер, существует только до первой вершины. Таким образом, A_{10} равно 0, и $A_{i0} = +\infty$ для всех i от 1 до $|V|$.

Теперь рассмотрим все пути из первой вершины в i, содержащие ровно j рёбер. Каждый такой путь есть путь из $j-1$ ребра, к которому добавлено последнее ребро. Если про пути длины $j-1$ все данные уже подсчитаны, то определить j-й столбец матрицы не составляет труда.

На псевдокоде алгоритм может быть записан так:

```
for v ∈ V do
    for i ← 0 to |V| − 1 do
        A_vi ← +∞
    end for
end for
A_10 ← 0
for i ← 1 to |V| − 1 do
    for (u, v) ∈ E do
        if A_vi > A_u,i−1 + w(u, v) then
            A_vi ← A_u,i−1 + w(u, v)
        end if
    end for
    for j ← 1 to |V| do
        Print(A_ji)
    end for
end for
```

Здесь V — множество вершин графа G, E — множество его рёбер, а $w(u, v)$ — вес ребра из вершины v в вершину u.

Вам задаётся количество вершин в графе и вывод этого алгоритма, необходимо построить какой либо граф, вывод этого алгоритма на котором будет совпадать с данным вам.

В первой строке входного файла задается число N — количество блоков. Каждое описание блока состоит из числа $|V|$ — количества вершин в графе и матрицы из $|V|-1$ строки по $|V|$ чисел — вывода алгоритма Беллмана-Форда. Бесконечность обозначается словом "inf".

В качестве ответа необходимо вывести N описаний графа, по одному для каждого блока. Каждое описание графа должно состоять из числа $|E|$ — количества рёбер в графе и $|E|$ троёк чисел u, v, w, описывающих рёбра, где u — номер вершины из которой исходит ребро, v — номер вершины в которую входит ребро, а w — вес этого ребра. Вес ребра должен быть целым и по модулю не должен превышать 10^8, в графе должны отсутствовать петли и мультирёбра (то есть ни для какого ребра $u \neq v$ и нет пары ребёр, где u и v совпадают).

В первом тесте $N = 3$. Оценка за этот тест: 30 баллов. Каждый верно построенный граф, оценивается в 10 баллов. Проверка осуществляется в режиме on-line (результат виден сразу).

Во втором тесте $N = 7$. Оценка за этот тест: 70 баллов. Каждый верно построенный граф, оценивается в 10 баллов. Во время тура проверяется, что сданный файл содержит описание 7 графов, а также отсутствие в графах петель и мультирёбер. Проверка правильности ответа осуществляется в режиме off-line (результат виден после окончания тура).

Примеры

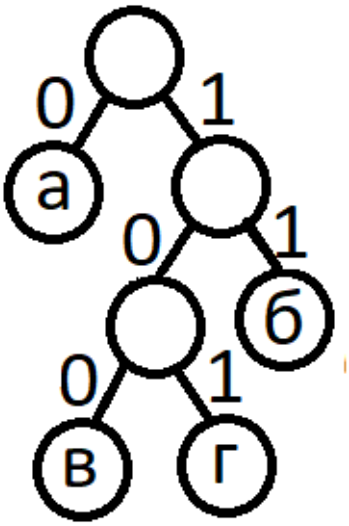
Входные данные	Результат
2 2 inf 1 3 inf 1 1 2 3 inf	1 1 2 1 5 1 2 1 1 3 1 2 1 1 3 2 2 2 1 1

Задача F. Дерево Хаффмана

Алгоритм Хаффмана позволяет кодировать символы алфавита беспрефиксным кодом различной длины, сопоставляя частым символам более короткий код, а редким — более длинный. Этот алгоритм используется во многих программах сжатия данных. Код символа определяется по следующим правилам:

1. Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, который может быть равен либо вероятности, либо количеству вхождений символа в сжимаемое сообщение.
2. Выбираются два свободных узла дерева с наименьшими весами.
3. Создается их родитель с весом, равным их суммарному весу.
4. Родитель добавляется в список свободных узлов, а два его потомка удаляются из этого списка.
5. Правой дуге, выходящей из родителя, ставится в соответствие бит 1, левой — бит 0. Битовые значения ветвей, исходящих от корня, не зависят от весов потомков.
6. Шаги, начиная со второго, повторяются до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.

Пусть буква «а» встречается в сообщении 4 раза, буква «б» — 3 раза, а буквы «в» и «г» — по 1 разу. Этим частотам может быть сопоставлено такое дерево:



Двоичный код буквы — это все цифры на пути из корня дерева в лист, соответствующей этой букве.

Для эффективного сжатия также важно максимально экономно хранить дерево Хаффмана. Опишем обход в глубину этого дерева, при этом мы будем сначала полностью обходить левое поддерево, затем возвращаться в узел, а затем обходить правое поддерево. Каждый раз проходя по ребру будем записывать букву L, R или U в зависимости от того, куда мы шли по ребру (L — в левого ребенка, R — в правого ребенка, U — в родителя). Приведенному в примере дереву будет соответствовать строка:

LURLLURUURUU

Такая строка позволяет однозначно восстановить дерево и сопоставить двоичные коды всем листьям дерева. Однако, запись можно модифицировать, заменив ребра типа L и R на ребра типа D, которое означает, что мы спускаемся в ребенка (сначала в левого, а если левый посещен — в правого). Тогда запись для нашего дерева будет выглядеть так:

DUDDDUDUUDUU

По этой строке также однозначно возможно восстановить структуру дерева. Она использует алфавит только из двух символов вместо трёх и может быть закодирована меньшим числом бит.

Эту запись также можно модифицировать, заменив смысл команды U. Если мы находимся в левом ребенке вершины, то в результате выполнения команды U мы поднимемся на один уровень и перейдем в правого ребенка вершины. Если же мы находимся в правом ребенке, то в результате применения команды U мы будем переходить к предку текущей вершины до тех пор, пока мы не станем левым ребенком (и сразу же поднимемся в предка и перейдём к правому ребенку, как это описано в предыдущем предложении). Запись для нашего дерева будет выглядеть так:

DUDDUU

Вам необходимо по записи, построенной по таким правилам, определить коды для всех листьев в порядке их обхода.

В первой строке входного файла задается число N — количество строк. Каждая из следующих N строк содержит описание обхода дерева.

В качестве ответа необходимо вывести N блоков кодов для каждой из строк входного файла. Каждый блок состоит из числа листьев K в этом дереве и из K строк, содержащих цифры 0 и 1 и описывающих код каждого из листьев.

В первом тесте $N = 3$. Оценка за этот тест: 30 баллов. Каждое дерево, для которого правильно определены коды листьев, оценивается в 10 баллов. Проверка осуществляется в режиме on-line (результат виден сразу).

Во втором тесте $N = 70$. Оценка за этот тест: 70 баллов. Каждое дерево, для которого правильно определены коды листьев, оценивается в 1 балл. Во время тура проверяется, что сданный файл содержит описание 70 блоков. Проверка правильности ответа осуществляется в режиме off-line (результат виден после окончания тура).

Примеры

Входные данные	Результат
2 DUDDUU DU	4 0 100 101 11 2 0 1

Задача G. Большая стройка

На одной из сторон нового проспекта начинается большая стройка. Было решено проводить строительство в точках, расположенных вдоль проспекта на равном расстоянии друг от друга.

Всего планируется построить H домов и P автобусных остановок (в каждой точке может быть расположен либо дом, либо остановка).

Чем ближе к дому остановка, тем удобнее жителям пользоваться общественным транспортом. Чем больше расстояние от дома до ближайшей остановки, тем меньшей популярностью он пользуется. Помогите проектировщикам по известным числам H и P определить, насколько будет удалён от остановки наименее популярный дом при оптимальной застройке.

В первой строке входного файла задается число N — количество планов застройки. Каждая из следующих N строк содержит по два числа H и P.

В качестве ответа необходимо вывести N чисел — расстояние от наименее популярного дома до ближайшей остановки для каждого из планов.

В первом тесте $N = 3$. Оценка за этот тест: 30 баллов. Каждое правильно вычисленное расстояние оценивается в 10 баллов. Проверка осуществляется в режиме on-line (результат виден сразу).

Во втором тесте $N = 70$. Оценка за этот тест: 70 баллов. Каждое правильно вычисленное расстояние оценивается в 1 балл. Во время тура проверяется, что сданный файл содержит 70 чисел. Проверка правильности ответа осуществляется в режиме off-line (результат виден после окончания тура).

Примеры

Входные данные	Результат
3 1 100 5 2 8 2	1 2 2