Технокубок 2017: Условия и разбор задач 3 отборочного раунда

A. Санта-Клаус и место в классе

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

ввод: стандартный ввод

вывод: стандартный вывод

Санта-Клаус пришел на рождественскую олимпиаду раньше всех, и ему предстоит первым занять свое место за партой! В аудитории, где находится его место, находятся n рядов по m парт, при этом за каждой партой находятся два рабочих места. Ряды пронумерованы слева направо от 1 до n, а парты в ряду — 1 до m начиная от доски. Обратите внимание, что в отличии от театров и кинотеатров ряды идут не параллельно экрану/доске, а перпендикулярно (см. рисунок).

Организаторы рождественской олимпиады пронумеровали все рабочие места числами от 1 до $2nm$. Места пронумерованы по рядам (т. е. сначала пронумерованы все места первого ряда, затем — все места второго и так далее), в одном ряду места пронумерованы начиная от доски (т. е. с первой парты этого ряда), за одной партой сначала пронумеровано место слева, затем — место справа.

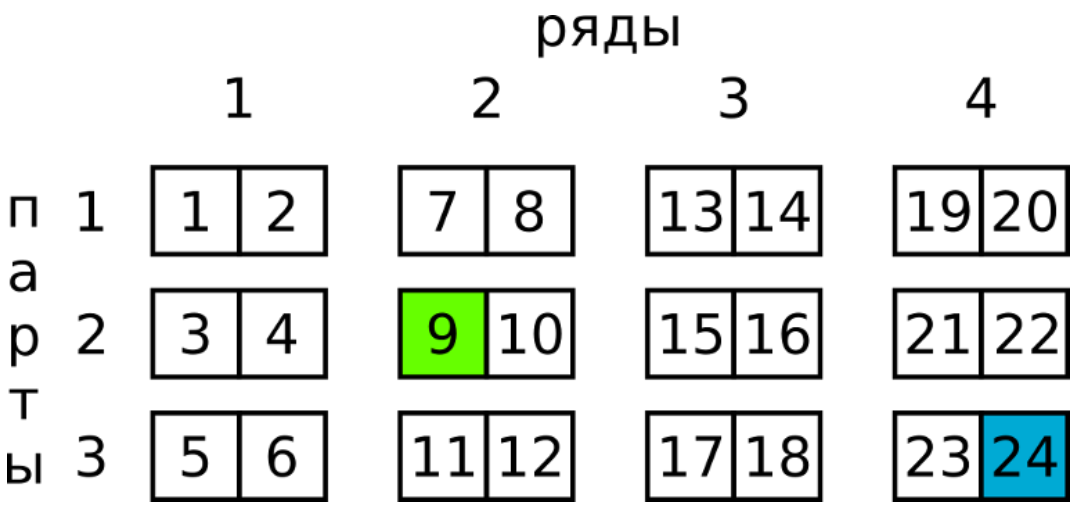


Рисунок иллюстрирует первый и второй примеры входных/выходных данных.

Санта-Клаус знает, что его место имеет номер k. Помогите ему узнать, в каком ряду за какую парту ему нужно сесть, а также слева его место или справа!

Входные данные

В единственной строке находятся три целых числа n, m и k ($1 \le n, m \le 10\,000$, $1 \le k \le 2nm$) — число рядов и число парт в каждом ряду, а так же номер места Санта-Клауса.

Выходные данные

Выведите два целых числа: номер ряда r, номер парты d, а также символ s, означающий сторону парты, за которую нужно сесть Санта-Клаусу. Символ s должен быть «L», если ему нужно сесть за левую сторону, и «R», если место Санта-Клауса справа.

Примеры

входные данные

```
4 3 9
```

выходные данные

```
2 2 L
```

входные данные

```
4 3 24
```

выходные данные

```
4 3 R
```

входные данные

```
2 4 4
```

выходные данные

```
1 2 R
```

Примечание

Первый и второй примеры показаны на картинке. Зеленым выделено место Санта-Клауса в первом примере, а голубым — его место во втором примере.

В третьем примере два ряда по четыре парты, а место Санта-Клауса — четвертое. Значит, его место в первом ряду за второй партой справа.

Решение:

Несложно заметить, что справа место Санта-Клауса, или слева, зависит только от четности *k*, а номер ряда и номер парты — только от $p = \lfloor \frac{k-1}{2} \rfloor$.

Видно, что при такой нумерации номер ряда равен $\lfloor \frac{p}{m} \rfloor + 1$, а номер парты $p \bmod m + 1$.

748B - Санта-Клаус и проверка клавиатуры

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

ввод: стандартный ввод

вывод: стандартный вывод

Санта-Клаус решил разобрать свою клавиатуру, чтобы её почистить. После того, как он поставил все клавиши обратно, он с ужасом понял, что что-то не так: некоторые пары клавиш перепутаны между собой! Таким образом, Санта-Клаус подозревает, что каждая клавиша либо стоит на своём месте, либо заняла место другой, а та другая — на месте первой.

Для того, чтобы убедиться в этом, найти ошибку и восстановить верное расположение, Санта-Клаус набрал текст своей любимой скороговорки, смотря только на надписи на клавиатуре.

Вам даны любимая скороговорка Санта-Клауса и строка, которая получилась в результате набора. Определите, какие пары клавиш Санта-Клаус мог перепутать. Каждая клавиша должна принадлежать **не более чем одной паре** перепутанных клавиш.

Входные данные

Входные данные состоят из двух строк *s* и *t* — любимой скороговорки Санта-Клауса и строки, которая получилась после набора скороговорки. Строки *s* и *t* непусты и имеют одинаковую длину, которая не превышает 1000, включительно. Обе строки состоят только из строчных латинских букв.

Выходные данные

Если предположение Санта-Клауса неверно и клавиатура требует починки и её нельзя починить, поменяв местами буквы в нескольких непересекающихся парах, выведите одно число «-1» (без кавычек).

Иначе в первой строке выведите число k ($k \geq 0$) — количество пар букв, которые нужно поменять местами. Затем в следующих kстроках выведите по две буквы, разделённые пробелом — буквы, которые необходимо поменять местами на клавиатуре. Все выведенные буквы должны быть различны.

Если ответов несколько, выведите любой. Как пары, так и буквы в парах можно выводить в любом порядке.

Каждая буква должна присутствовать не более чем в одной паре. Санта-Клаус считает, что клавиши расположены корректно, если он может набрать на клавиатуре текст своей любимой скороговорки без ошибок.

Примеры

входные данные

```
helloworld
ehoolwlroz
```

выходные данные

```
3
h e
l o
d z
```

входные данные

```
hastalavistababy
hastalavistababy
```

выходные данные

```
0
```

входные данные

```
merrychristmas
christmasmerry
```

выходные данные

```
-1
```

Решение:

Обозначим две имеющиеся строки через s и t. Достаточно найти все пары различных символов s_i и t_i и сделать следующее:

- Убедиться, что каждый символ встречается не более, чем в одной различной паре,

- Убедиться, что если символ c в паре с некоторым символом d, то всякому вхождению c в s на i-й позиции соответствует $t_i = d$, и наоборот.

Если хотя бы одно из этих условий не выполнено, то починить клавиатуру невозможно, иначе достаточно вывести полученные пары.

748C - Санта-Клаус и его Робот

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

У Санта-Клауса есть Робот, который живёт на клетчатой плоскости и умеет перемещаться **по линиям сетки**. Если ему дать последовательность из m точек $p_1, p_2, ..., p_m$ с целыми координатами, то он сделает следующее: обозначим точку, в которой он сейчас находится, через p_0. Тогда Робот сначала поедет по некоторому кратчайшему пути из p_0 в p_1 (обратите внимание, поскольку Робот ездит только по линиям сетки, кратчайших путей может быть несколько), затем, доехав до p_1, поедет к точке p_2, опять же, по какому-то кратчайшему пути, затем к точке p_3, и так далее, пока не пройдёт все точки в заданном порядке. Некоторые из точек в последовательности могут совпадать, тогда Санта-Клаус должен посетить их несколько раз в порядке, соответствующем последовательности.

Пока Санта-Клауса не было, кто-то дал Роботу несколько точек. Эта последовательность точек была утерян, но у вас есть протокол перемещений Робота (каждое перемещение на единицу длины). Узнайте, пожалуйста, минимальную возможную длину последовательности, заданной Роботу.

Входные данные

В первой строке задано единственное натуральное число n ($1 \le n \le 2 \cdot 10^5$) — число перемещений Робота на единичный отрезок. Во второй строке дан протокол перемещений Робота в виде n символов, каждый из которых равен L, R, U или D, записанных без пробелов. k-й символ означает, что на k-м шаге Робот переместился на единицу длины в направлении, соответствующем этому символу: L означает, что он двигался влево, R — вправо, U — вверх и D — вниз. Смотрите иллюстрации к примерам для большего понимания.

Выходные данные

В единственной строке выведите минимальную возможную длину последовательности, заданной Роботу.

Примеры

входные данные

```
4
RURD
```

выходные данные

```
2
```

входные данные

```
6
RRULDD
```

выходные данные

```
2
```

входные данные

```
26
RRRULURURUULULLLDLDDRDRDLD
```

выходные данные

```
7
```

входные данные

```
3
RLL
```

выходные данные

```
2
```

входные данные

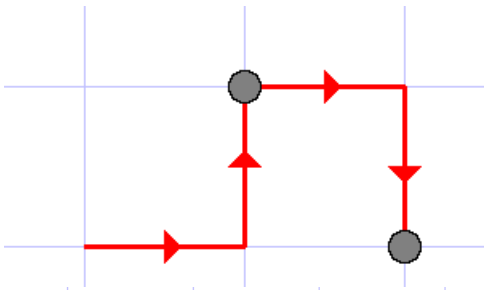
```
4
LRLR
```

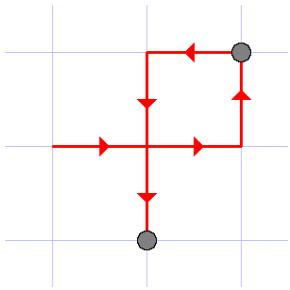
выходные данные

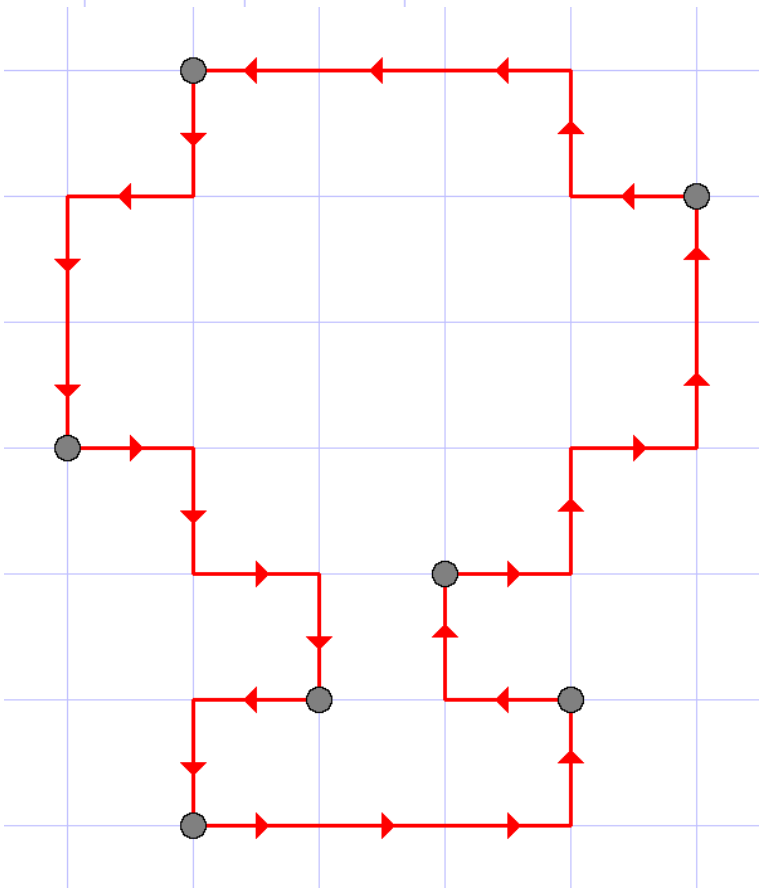
```
4
```

Примечание

Ниже приведены иллюстрации к первым трём тестам.







Последний пример показывает, что каждая точка в последовательности должна быть посчитана столько раз, сколько она в ней встречается.

Решение 1: обозначим через dp_i ответ на задачу, если бы робот прошёл только i единичных отрезков, dp_0 положим равным нулю. Таким образом, ответом на задачу является dp_n. Легко видеть, что $dp_{k+1} \geq dp_k$ для всех k, поэтому для любого $i \leq n$верно $dp_i = dp_j + 1$, где j есть минимальный возможный индекс, для которого выполняется, что $s[j + 1... i]$ может быть кратчайшим путём между двумя точками. Это эквивалентно выполнению следующих двух условий:

- $s[j + 1... i]$ не содержит или L, или R (возможно, ни того, ни другого);
- $s[j + 1... i]$ не содержит или U, или D.

(доказательство этого факта предоставляется читателю в качестве упражнения).

Такое j можно найти за $O(1)$ времени, проходясь i от 1 до n и запоминая моменты последнего вхождения букв L, R, U и D. Таким образом, запоминая эти величины и массив dp, можно выполнить вышеприведённый алгоритм за $O(n)$ времени.

Решение 2: поскольку мы уже знаем, как понять по пути, может ли он быть кратчайшим путём между двумя точками (из первого решения), найдём наидлиннейший префикс s, удовлетворяющий этим условиям, и скажем, что это кратчайший путь до первой точки. Затем мы найдём длиннейший подходящий префикс для оставшейся строки, и скажем, что это путь от p_1 до p_2, и так далее.

Этот жадный алгоритм работает (очевидно, за линейное время), потому что первое решение на самом деле выполняет тот же алгоритм, применённый к развёрнутой строке.

748D - Санта-Клаус и палиндром

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

ввод: стандартный ввод

вывод: стандартный вывод

Санта-Клаус очень любит палиндромы. Недавно у него был день рождения. В гости к Санта-Клаусу пришли k друзей, каждый из них подарил ему строку s_i одной и той же длины n, красота i-й строки равна a_i. Возможно, что a_i является отрицательным — это значит, что строка не является красивой по мнению Санта-Клауса.

Санта-Клаус без ума от палиндромов. Ему стало интересно: какую максимальную суммарную красоту может иметь палиндром, если склеить какие-то (возможно все) из подаренных строк? Каждый подарок можно использовать не более одного раза. Обратите внимание, что все подаренные строки имеют **одинаковую длину** n.

Напоминаем, что палиндром — это строка, которая не изменится, если её развернуть задом наперёд.

Так как пустая строка является палиндромом, то искомая максимальная красота — неотрицательна. Даже если все a_iотрицательны, Санта-Клаус может получить пустую строку в качестве палиндрома.

Входные данные

В первой строке задано два целых числа через пробел, k и n — количество друзей Санта-Клауса и длина строки, подаренной каждым другом ($1 \le k, n \le 100\,000$; $n \cdot k \le 100\,000$).

Далее следуют k строк. i-я из них содержит подаренную строку s_i и её красоту a_i ($-10\,000 \le a_i \le 10\,000$). Строка состоит из nстрочных букв

латинского алфавита, а её красота — целое число. Подаренные строки могут совпадать. Одинаковые строки могут иметь разную красоту.

Выходные данные

Выведите искомую максимальную суммарную красоту.

Примеры

входные данные

```
7 3
abb 2
aaa -3
bba -1
zyz -4
abb 5
aaa 7
xyx 4
```

выходные данные

```
12
```

входные данные

```
3 1
a 1
a 2
a 3
```

выходные данные

```
6
```

входные данные

```
2 5
abcde 10000
abcde 10000
```

выходные данные

0

Примечание

В первом примере из условия Санта-Клаус может склеить палиндром abbaaaxuxaaabba, используя строки 5, 2, 7, 6, 3(склеив их именно в этом порядке).

Решение:

Представьте палиндром, разбитый на куски одинакового размера. Например,

ABC DEF XYX FED CBA

У каждого куска, кроме центрального, есть пара: та же самая строка, но развёрнутая. Так, здесь пара первого куска ABC — последний кусок CBA, пара второго — предпоследний.

Предположим, что в ответ войдёт чётное число строк. Таким образом у каждой будет пара, потому что центральной строки нет. Разобьём все строки на группы; S и $rev(S)$ отнесём к одной группе. Теперь, если S — не палиндром, мы можем сделать пару из самого дорогого вхождения S и самого дорогого вхождения $rev(S)$, повторяя столько раз, пока есть необходимые строки и пока сумма стоимостей положительна. Если же S — палиндром, на каждом шаге нужно брать два вхождения Sмаксимальной стоимости (опять же, до тех пор, пока есть хотя бы два вхождения и сумма их стоимостей положительна).

Основная сложность в центральном блоке. Очевидно, там могут стоять только строки-палиндромы. Однако, их может быть много, и нам нужно выбрать такую, которая максимизирует суммарный ответ. Нужно разобрать пару случаев. Посмотрим на момент, когда мы взяли последнюю пару среди строк S и остановились.

Если значение следующего вхождения положительно (пусть x), можно «положить его в карман» и сказать, что x — *потенциальный бонус* за то, что строка S будет в центре. Иначе может быть нужно разбить последнюю пару. Пусть стоимости последней пары, соответственно, a и b ($a \geq b$). Заметим, что $b < 0$, потому что иначе нет смысла использовать первый элемент отдельно. Теперь, если мы возьмём пару целиком, то получим $a + b$ очков, если же только первый элемент, то a очков. Значит, *потенциальный бонус* в этом случае равен $-b$. Обратите внимание, что $a + b$, набранные парой, всё ещё необходимо учесть в основном ответе!

Теперь нужно взять строку с максимальным потенциальным бонусом, поставить её в центр и прибавить значение бонуса к ответу.

748E - Санта-Клаус и мандарины

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 256 мегабайт

ввод: стандартный ввод

вывод: стандартный вывод

У Санта-Клауса есть n мандаринов, причём i-й из них имеет a_i долек. Санта-Клаус пришел в школу, где учится k учеников. Он решил угостить их мандаринами.

Так как на всех мандаринов может не хватить, Санта-Клаус решил поделить их на части, чтобы никто не обиделся. Для этого он может делить мандарины пополам, а также делить любую часть пополам. Если количество долек в мандарине или в части, которую Санта-Клаус делит, нечётное, то в одной получившейся части окажется на одну дольку больше, чем в другой. Делить

мандарин или часть мандарина можно лишь в том случае, если после деления каждая получившаяся часть будет иметь хотя бы одну дольку.

Санта-Клаус хочет дать каждому из школьников ровно один мандарин или одну часть мандарина (в частности, каждый ученик должен получить положительное количество долек). Возможно, что у Санта-Клауса останется несколько мандаринов или частей после того, как он раздаст часть из них школьникам.

Пусть в результате угощения i-му ученику достанется b_i долек. В таком случае *радость* Санта-Клауса будет равна минимальному значению среди всех b_i.

Найдите максимальную возможную величину *радости* Санта-Клауса, которую он может получить после угощения учеников мандаринами.

Входные данные

В первой строке находятся два целых положительных числа n и k ($1 \le n \le 10^6$, $1 \le k \le 2 \cdot 10^9$) — количество мандаринов и количество учеников.

Во второй строке находится последовательность из n целых чисел $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^7$), где a_i равно количеству долек в i-м мандарине.

Выходные данные

Если невозможно раздать всем ученикам по манадарину или части мандарина, выведите -1. В противном случае выведите максимально возможную величину *радости* Санта-Клауса.

Примеры

входные данные

```
3 2
5 9 3
```

выходные данные

```
5
```

входные данные

```
2 4
12 14
```

выходные данные

```
6
```

входные данные

```
2 3
1 1
```

выходные данные

```
-1
```

Примечание

В первом примере Санта-Клаусу нужно разделить второй манадрин пополам, чтобы в одной части было 5 долек, а в другой 4. Тогда он сможет отдать одному ученику часть, в которой 5 долек, а второму целый первый манадарин, в котором также 5долек.

Во втором примере Санта-Клаусу нужно разделить пополам оба манадрина, тогда он сможет отдать двум ученикам части по 6долек, а двум другим ученикам — части по 7 долек.

В третьем примере Санта-Клаус не сможет дать всем ученикам хотя бы по одной дольке, так как у него есть всего 2 дольки и 3ученика.

Решение:

Понятно, что если суммарно долек меньше, чем k, то ответ - 1. Иначе ответ хотя бы 1. Найдем его.

Будем последовательно делить мандарины и части пополам в поисках лучшего ответа. Понято, что делить часть размера xневыгодно, пока есть неразделенная часть размера $y > x$. Поэтому будем делить мандарины, на каждом шаге деля самую большую часть, и будем поддерживать ответ, а

именно размер k-й наибольшей части, а также множество частей, которые мы раздадим детям. При делении наибольшей дольки на две возможно два случая:

- Если размер какой-либо из получившихся частей меньше, чем текущий ответ, то понятно, что в дальнейшем ответ не будет улучшаться, а значит можно прекращать процесс.
- Если размер обеих получившихся частей не меньше, чем текущий ответ, то надо из текущего множества частей, которые мы раздадим, убрать наименьшую часть, таким образом добившись, что во множестве снова ровно k частей.

Видно, что во втором случае ответ всегда не ухудшается, а значит можно просто повторять операцию разделения наибольшей части до тех пор, пока не реализуется первый случай.

Чтобы быстро реализовывать этот процесс, можно хранить в массиве от 1 до A, где $A = 10^7$ — максимально возможный размер части, текущее количество частей с таким количеством долек. Тогда, если поддерживать два указателя на текущий ответ и на текущий максимальный размер части, можно выполнить весь процесс за $O(A)$. Итоговая асимптотика $O(n + A)$.

748F - Санта-Клаусы и чемпионат по футболу

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

В Древляндии n городов, соединенных между собой n - 1 дорогой так, что из каждого города можно по дорогам добраться до любого другого. В

следующем году в Древляндии пройдет чемпионат по футболу среди команд, состоящих из Санта-Клаусов. Всего в чемпионате примут участие $2k$ команд из $2k$ различных городов.

На первом этапе команды разобьются на k пар, каждая пара сыграет два матча: один в городе первой команды из пары, другой — в городе второй команды из пары. Таким образом, в каждом из $2k$ городов участников будет проведен ровно один матч. Однако как разбить команды на пары, пока не решено.

Перед организаторами также стоит задача определить, в каких городах расселить участников на время чемпионата. Организаторы хотят поселить команды в как можно **меньшее** число городов, чтобы Санта-Клаусы побольше общались друг с другом и обменивались опытом.

Никакая из команд не хочет чересчур много передвигаться во время турнира, поэтому если команда будет играть в городах u и v, то жить она хочет в городе, лежащем на кратчайшем пути из u в v (в том числе, возможно, в городе u или городе v). Также команды из одной пары необходимо расселить в одном городе.

Таким образом, организаторы турнира хотят разбить $2k$ команд на пары и расселить участников по минимально возможному числу m городов так, чтобы команды из одной пары жили в одном городе, и чтобы для каждой команды город, в котором она будет жить во время турнира, лежал на кратчайшем пути между городами, в которых она будет играть.

Входные данные

В первой строке заданы два целых числа n и k ($2 \le n \le 2 \cdot 10^5$, $2 \le 2k \le n$) — количество городов в Древляндии и количество пар команд в турнире.

Следующие n - 1 строк задают описание дорог в Древляндии, каждая из них содержит два целых числа a, b ($1 \le a, b \le n$, $a \ne b$), что значит, что

города a и b соединены очередной дорогой. Гарантируется, что из любого города можно добраться в любой другой по дорогам.

Следующая строка содержит $2k$ различных целых чисел $c_1, c_2, \ldots, c_{2k}$ ($1 \le c_i \le n$), где c_i — город, из которого команда номер i. Все эти числа различны.

Выходные данные

В первой строке выведите целое число m — минимальное количество городов, по которым можно расселить участников.

Во второй строке выведите m различных чисел чисел $d_1, d_2, \ldots, d_m$ ($1 \le d_i \le n$) — номера городов, в которых будут жить участники.

Далее выведите k строк. В j-й из них выведите 3 числа u_j, v_j, x_j, где u_j и v_j — города, в которых будет играть j-я пара, а x_j — номер города, в котором будут жить команды этой пары. Каждое из чисел $c_1, c_2, \ldots, c_{2k}$ должно встретиться среди чисел u_j и v_j ровно один раз. Каждое из чисел x_j должно принадлежать множеству $\{d_1, d_2, \ldots, d_m\}$.

Если оптимальных ответов несколько, выведите любой из них.

Пример

входные данные

```
6 2
1 2
1 3
2 4
2 5
3 6
2 5 4 6
```

выходные данные

```
1
2
```

```
5 4 2
6 2 2
```

Примечание

В первом тесте из условия можно расселить всех участников в один город с номером 2. Разбить на пары можно любым образом, при этом все условия всегда будут выполнены, т. к. город 2 лежит на кратчайшем пути между любой парой городов из множества {2, 4, 5, 6}.

Решение:

Сначала докажем, что существует вершина v такая, что все поддеревья ее соседей, не содержащие v, содержат не более k вершин, в которых будут сыграны матчи (дальше мы будем называть такие вершины *выбранными*). Подвесим дерево за какую-нибудь вершину *root* и обозначим $f(u)$ количество выбранных вершин в поддереве u. Теперь нужно доказать, что существует вершина v такая, что $f(v) \geq k$ (тогда не больше k выбранных вершин лежат в дополнении к поддереву v), но для всех детей вершины v to выполнено $f(to) \leq k$. Пусть нет искомой вершины v, тогда для всех вершин u таких, что $f(u) > k$, существует сын u' такой, что $f(u') > k$. Тогда можно построить последовательность $u_0, u_1, \ldots$, в которой $u_0 = root$, u_{i+1} — сын u_i и выполнено $f(u_i) > k$. Очевидно, в какой-то момент последовательность придет в лист, в котором $f(u) \leq 1$. Получили противоречие с тем, что $f(u) > k$.

Это доказательство также дает алгоритм нахождения v: посчитаем $f(u)$ для всех поддеревьев и спустимся от корня, всегда переходя в сына с $f(u_{i+1}) > k$. В какой-то момент такого сына не будет, что будет означать, что мы находимся в искомой вершине.

Используя доказанное утверждение, нетрудно показать, что всегда можно поселить все команды в одну вершину. Рассмотрим вершину v такую,

что $f(to) \le k$ для всех ее соседей. Построим список выбранных вершин, сначала выпишем все выбранные вершины из поддерева первого сына v, потом выбранные вершниы из поддерева второго сына v, и так далее. Если v тоже выбрана, допишем ее в конец списка. Получили список из $2k$ вершин a. Разобьем их на k пар вида (a_i, a_{i+k}). Поскольку для всех детей v выполнено $f(to) \le k$, то вершины, чьи позиции отличаются в списке на k, не могут принадлежать одному поддереву (так как выбранные вершины в каждом из поддеревьев мы выписывали подряд). Поэтому кратчайшй путь между a_i и a_{i+k} будет проходить через вершину v.