

Задача А. Декартово

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Государство Иксово состоит из N_x городов, некоторые пары которых связаны дорогами с двусторонним движением. Каждая дорога имеет свою длину. Всего межгородских дорог в стране M_x , причем известно, что из каждого города Иксевщины можно доехать по дорогам до каждого другого города этой страны. Города Иксово пронумерованы натуральными числами от 1 до N_x .

Государство Игреково состоит из N_y городов, некоторые пары которых связаны дорогами с двусторонним движением. Каждая дорога имеет свою длину. Всего межгородских дорог в стране M_y , причем известно, что из каждого города Игреково можно доехать по дорогам до каждого другого города этой страны. Города Игреково пронумерованы натуральными числами от 1 до N_y .

Страна Декартово состоит из $N = N_x \cdot N_y$ городов: каждому городу Декартово во взаимно однозначное соответствие можно поставить пару городов-побратимов (x, y) , где x — город Иксово, а y — город Игреково. Некоторые пары городов Декартово также соединены дорогами с двусторонним движением. Дорог в стране ровно $M = N_x \cdot M_y + N_y \cdot M_x$. При этом дорога между городами (x_1, y_1) и (x_2, y_2) существует только в одном из таких двух случаев:

1. Если $x_1 = x_2$, а между городами y_1 и y_2 Игреково проложена дорога. При этом длина дороги между городами (x, y_1) и (x, y_2) Декартово равно длине дороги между городами y_1 и y_2 Игреково.
2. Если $y_1 = y_2$, а между городами x_1 и x_2 Иксевщины проложена дорога. При этом длина дороги между городами (x_1, y) и (x_2, y) Декартово равно длине дороги между городами x_1 и x_2 Иксево.

Города разных государств между собой дорогами не соединены.

Данная задача состоит из двух подзадач. В обеих подзадачах всю информацию про соединение дорогами задано во входных файлах.

В первой подзадаче требуется определить длину самого короткого пути по дорогам Декартовщины из города $(1, 1)$ в город (N_x, N_y) .

Во второй подзадаче некоторые дороги Декартовщины требуется закрыть. Ваша задача — определить, дороги какой наименьшей суммарной длины можно оставить в Декартовщине, чтобы из любого ее города все еще можно было попасть в любой другой.

Формат входных данных

Первая строка входного файла содержит номер подзадачи, которую требуется решить (1 или 2). Вторая строка содержит натуральные числа N_x и M_x ($1 \leq N_x, M_x \leq 5 \cdot 10^4$) — количество городов и дорог в Иксово. В последующих M_x строках описаны дороги Иксово: в каждой строке по три числа, где первые два задают номера разных городов, соединенных дорогой, а третья есть длиной соответствующей дороги (натуральное число, которое не превышает 10^7).

В следующей строке входного файла указаны натуральные числа N_y и M_y ($1 \leq N_y, M_y \leq 5 \cdot 10^4$) — количество городов и дорог в Игреково. Последующие M_y строк содержат описание дорог Игреково; формат данных и ограничения соответствуют описанным выше.

Формат выходных данных

Выведите единственное целое число — ответ на вопрос подзадачи.

Примеры

стандартный ввод	стандартный вывод
1 3 2 2 1 15 3 1 14 3 2 2 1 15 3 2 15	44
2 3 2 2 1 15 3 1 14 3 2 2 1 15 3 2 15	117

Задача В. Расстояние X

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Вам дан неориентированный взвешенный граф из N вершин и M ребер. Назовем ценой пути между двумя вершинами вес максимального ребра на этом пути. Найдите количество пар с минимальной ценой пути между ними, равной X .

Формат входных данных

Первая строка содержит три целых числа N , M и X ($1 \leq N \leq 10^5, 1 \leq M \leq 3 \cdot 10^5, 1 \leq X \leq 10^9$). Следующие M строк содержат по три числа a_i , b_i и w_i , обозначающие ребро между вершинами a_i и b_i веса w_i . ($1 \leq w_i \leq 10^9$)

Формат выходных данных

Выведите одно число — ответ на задачу

Примеры

стандартный ввод	стандартный вывод
7 6 3 1 2 1 1 3 2 3 4 3 4 5 1 4 6 2 1 7 4	9
8 8 4 1 3 2 2 4 1 1 5 1 6 7 3 5 8 4 8 4 4 6 5 5 7 8 6	11

Задача С. Всем чмоки в этом чатике!

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Сегодня Мэри, как программисту социальной сети «Телеграфчик», предстоит реализовать сложную систему управления чатами.

Задача Мэри усложняется тем, что в социальную сеть «Телеграфчик» внедрена продвинутая система шифрования «ZergRus», простая, как всё гениальное. Суть её в том, что в системе хранится одна переменная $zerg$, которая принимает значения от 0 (включительно) до $p = 10^6 + 3$ (исключая p) и меняется в зависимости от событий в системе.

В социальной сети всего n пользователей ($1 \leq n \leq 10^5$). В начале дня каждый пользователь оказывается в своём собственном чате, в котором больше никого нет. Переменная $zerg$ в начале дня устанавливается равной 0.

В течение дня происходят события типов:

1. Участник с номером $(i + zerg) \bmod n$ посылает сообщение всем участникам, сидящим с ним в чате (в том числе и себе самому), при этом переменная $zerg$ заменяется на $(30 \cdot zerg + 239) \bmod p$.
2. Происходит слияние чатов, в которых сидят участники с номерами $(i + zerg) \bmod n$ и $(j + zerg) \bmod n$. Если участники и так сидели в одном чате, то ничего не происходит. Если в разных, то чаты объединяются, а переменной $zerg$ присваивается значение $(13 \cdot zerg + 11) \bmod p$.
3. Участник с номером $(i + zerg) \bmod n$ хочет узнать, сколько сообщений он не прочитал, и прочитать их. Если участник прочитал q новых сообщений, то переменной $zerg$ присваивается значение $(100\,500 \cdot zerg + q) \bmod p$.

Вы поможете Мэри реализовать систему, обрабатывающую эти события?

Формат входных данных

В первой строке входного файла записаны натуральные числа n ($1 \leq n \leq 10^5$) — число пользователей социальной сети, и m ($1 \leq m \leq 3 \cdot 10^5$) — число событий, произошедших за день. В следующих m строках содержится описание событий. Первое целое число в строке означает тип события t ($1 \leq t \leq 3$). Если $t = 1$, далее следует число i ($0 \leq i < n$), по которому можно вычислить, какой участник послал сообщение. Если $t = 2$, далее следуют числа i и j ($0 \leq i, j < n$), по которым можно вычислить номера участников, чаты с которыми должны объединиться. Если $t = 3$, далее следует число i ($0 \leq i < n$), по которому можно вычислить номер участника, желающего узнать, сколько у него сообщений, и прочитать их.

Формат выходных данных

Для каждого события типа 3 нужно вывести число непрочитанных сообщений у участника.

Пример

стандартный ввод	стандартный вывод
4 10	1
1 0	1
1 2	2
1 1	
1 2	
3 1	
2 1 2	
1 3	
3 3	
2 3 2	
3 2	

Задача D. Еще одна задача про остов

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Костя с Ваней при подготовке очередной задачи на персистентную двумерную динамическую структуру данных нашли массив a_i размера n , состоящий из целых чисел, каждое из которых находится в промежутке $[0, m - 1]$. Костя, как любитель цветных графов, сразу создал полный неориентированный граф из n вершин, вершине v был задан цвет a_v . Ваня, как любитель взвешенных графов, сразу задал каждому ребру (v, u) вес $f(a_v, a_u)$.

$$f(a, b) = \begin{cases} a + b & a + b < m \\ a + b - m, & \text{иначе} \end{cases}$$

Теперь ребят интересует вопрос: каким же будет вес минимального остовного дерева для нашего графа? Помогите им найти ответ.

Формат входных данных

Программа получает на вход два числа n, m ($1 \leq n \leq 2 \cdot 10^5, 1 \leq m \leq 10^9$).

В следующей строке вводится n целых чисел — массив a_i ($0 \leq a_i \leq m - 1$).

Формат выходных данных

Выведите одно число — суммарный вес остовного дерева.

Пример

стандартный ввод	стандартный вывод
3 2 0 1 1	1

Задача Е. Разностный MST

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2 секунды
Ограничение по памяти:	1024 мегабайта

Дан массив x_1, x_2, \dots, x_n .

Давайте создадим неориентированный граф на n вершинах, в котором изначально нет ребер.

После этого для каждой пары (u, v) , такой что $u < v$, добавим в граф ребро между вершинами u и v веса $x_v - x_u$.

Найдите вес минимального остовного дерева в получившемся графе.

Формат входных данных

Первая строка входных данных содержит одно целое положительное число t ($1 \leq t \leq 300\,000$) — количество тестовых примеров.

Первая строка каждого тестового примера содержит одно целое положительное число n ($1 \leq n \leq 300\,000$) — количество элементов массива.

Вторая строка каждого тестового примера содержит n целых чисел x_1, x_2, \dots, x_n ($-300\,000 \leq x_i \leq 300\,000$) — элементы массива.

Гарантируется, что сумма n по всем тестовым примерам не превышает 300 000.

Формат выходных данных

Для каждого тестового примера выведите одно целое число — вес минимального остовного дерева в данном графе.

Пример

стандартный ввод	стандартный вывод
2	4
5	-35
1 2 3 4 5	
3	
10 45 10	

Задача F. Кукушки

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Британские учёные решили заняться орнитологией и понаблюдать за жизнью необычных кукушек. Для этого они вырастили дерево и построили на нём n гнёзд, в каждом из которых живёт кукушка. Наблюдение за деревом состоит в том, что в некоторые моменты времени учёные оценивают, можно ли подложить определённое яйцо в гнездо к некоторой кукушке или нет.

Каждое яйцо может вынашиваться только в двух определённых гнёздах. Каждое яйцо задаётся неупорядоченной парой различных чисел (x, y) . Яйцо (x, y) может вынашиваться в любом из гнёзд x и y и не может вынашиваться в других гнёздах. Обратите внимание, яйцо (x, y) не отличается от яйца (y, x) .

Теперь опишем процесс подкладывания яйца в имеющиеся гнезда: пусть учёные хотят подложить яйцо (x, y) в гнездо x . Если в гнезде x нет яйца, то яйцо (x, y) просто остаётся в этом гнезде, и процесс на данном шаге завершается. Если же в гнезде x лежит какое-то яйцо (x, p) , то кукушка кладёт яйцо (x, y) в данное гнездо, а яйцо (x, p) пытается подложить в гнездо p аналогичным образом, и процесс продолжается.

Вам предлагается отвечать на вопросы учёных. Всего есть три типа вопросов:

- (Теоретический) Закончится ли процесс, если подложить яйцо (x, y) в гнездо x ? Так как вопрос чисто теоретический, оно **не добавляется** на самом деле, и состояние гнёзд не меняется.
- (Практический) Закончится ли процесс, если подложить яйцо (x, y) в гнездо x ? Если процесс закончится, то яйцо **добавляется** в реальности согласно описанному процессу.
- (Теоретический) Сколько существует **упорядоченных** пар различных чисел (x, y) , таких что яйцо (x, y) можно подложить в гнездо x с учётом имеющихся в гнёздах яиц? При этом для каждого яйца ответ определяется независимо от других добавляемых яиц.

Формат входных данных

В первой строке вводятся три целых числа n, m, q , ($2 \leq n \leq 200\,000$, $0 \leq m \leq n$, $1 \leq q \leq 600\,000$), где n — количество гнёзд на дереве, m — количество яиц, которые учёные уже положили, q — количество вопросов, которые задают учёные.

В каждой из m последующих строк следуют по два числа x_i, y_i , означающих, что в гнезде x_i лежит яйцо (x_i, y_i) . Гарантируется, что все x_i различны и что $x_i \neq y_i$ для всех i .

В следующих q строках описаны вопросы учёных. Вопросы даны в том порядке, в котором на них требуется отвечать. Первое число t_j в строке описывает тип вопроса.

Если $t_j = 1$ или $t_j = 2$, то далее идут два различных числа x_j и y_j , описывающих яйцо, которое фигурирует в соответствующем вопросе.

Если $t_j = 1$, то яйцо не требуется добавлять в текущую расстановку.

Если $t_j = 2$, то яйцо требуется добавить, если процесс добавления потребует конечного числа перекладываний.

Если $t_j = 3$, то требуется определить количество упорядоченных пар (x, y) , таких что яйцо (x, y) можно добавить в гнездо x с тем, чтобы процесс когда-нибудь завершился. В реальности никакие яйца в расстановку не добавляются.

Формат выходных данных

Для каждого вопроса первого и второго типа выведите единственное слово «Yes» или «No» в зависимости от того, закончится ли процесс перекладывания.

Для каждого запроса третьего типа выведите количество искомых упорядоченных пар.

Система оценки

Тесты к этой задаче проходятся в случае прохождения всех тестов. Пусть t_1 — количество запросов, а t_2 и t_3 — количество запросов третьего и четвёртого уровня соответственно. Для каждой группы даны ограничения на количество запросов и время выполнения. Если группа не пройдена, то количество запросов и время выполнения будут равны 0.

Группа	Баллы	Дополнительные ограничения				Необх. группы	Комментарий
		n	t_1	t_2	t_3		
0	0	—	—	—	—	—	Пропущена
1	13	$n \leq 2000$	$t_1 \leq 2000$	$t_2 = 0$	$t_3 = 0$	—	
2	14	$n \leq 2000$	$t_1 \leq 2000$	$t_2 = 0$	$t_3 \leq 1$	1	
3	12	$n \leq 2000$	$t_1 \leq 2000$	$t_2 \leq 2000$	$t_3 \leq 2000$	0 – 2	
4	12	—	$t_1 \leq 2 \cdot 10^5$	$t_2 = 0$	$t_3 = 0$	1	
5	18	—	$t_1 \leq 2 \cdot 10^5$	$t_2 = 0$	$t_3 \leq 1$	1 – 2, 4	
6	31	—	$t_1 \leq 2 \cdot 10^5$	$t_2 \leq 2 \cdot 10^5$	$t_3 \leq 2 \cdot 10^5$	0 – 5	Offline-платформа

Пример

стандартный ввод	стандартный вывод
5 3 8	Yes
1 2	20
5 1	Yes
2 4	8
1 1 2	No
3	Yes
2 1 2	0
3	No
2 4 2	
2 5 3	
3	
1 4 5	

Замечание

Изначальное расположение яиц в тесте из условия такое: в первом гнезде лежит яйцо (1, 2), во втором — (2, 4), в пятом — (5, 1), а в третьем и четвёртом яиц нет.

Яйцо (1, 2) добавить можно, несмотря на то что подобное яйцо на дереве уже есть, это приведёт к перекладыванию имеющегося яйца (1, 2) в другое гнездо.

Также в начальную конфигурацию можно добавить любое из 10 яиц, существующих для дерева с пятью гнёздами, и каждое яйцо можно положить в любое из двух гнёзд, ему отвечающих, и для любого из добавляемых яиц и гнёзд это потребует конечное количество шагов. Таким образом, ответ на второй запрос — 20.

В результате следующего запроса яйцо (1, 2) будет добавлено реально, и распределение яиц будет таким: в первом гнезде лежит яйцо (1, 2), во втором — также (1, 2), в четвёртом — (2, 4), в пятом (5, 1).

Теперь уже можно добавить только яйца (1, 3), (2, 3), (4, 3) и (5, 3), причём по-прежнему любое яйцо можно положить в каждое из двух упомянутых на нём гнёзд, поэтому ответ на запрос — 8.

Яйцо (4, 2) добавить на дерево нельзя, поэтому состояние гнёзд не изменится.

Для добавления яйца (5, 3) понадобится 5 перекладываний яиц, а после этого никакое новое яйцо за конечное количество шагов добавить уже нельзя.

Задача G. Соединение и разъединение

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

Вы когда-нибудь слышали про обход в глубину? Например, используя этот алгоритм, вы можете проверить является ли граф связным за время $O(E)$. Вы можете даже посчитать количество компонент связности за то же время.

А вы когда-нибудь слышали про систему непересекающихся множеств? Используя эту структуру, вы можете быстро обрабатывать запросы “Добавить ребро в граф” и “Посчитать количество компонент связности в графе”.

А вы когда-нибудь слышали о *динамической* задаче связности? В этой задаче вам необходимо обрабатывать три типа запросов:

1. Добавить ребро в граф.
2. Удалить ребро из графа.
3. Посчитать количество компонент связности в графе.

Можно считать, что граф является неориентированным. Изначально граф является пустым.

Формат входных данных

В первой строке находятся два целых числа N и K — количество вершин и количество запросов, соответственно ($1 \leq N \leq 300\,000$, $0 \leq K \leq 300\,000$). Следующие K строк содержат запросы, по одному в строке. Каждый запрос имеет один из трех типов:

1. $+ u v$: Добавить ребро между вершинами u и v . Гарантируется, что такого ребра нет.
2. $- u v$: Удалить ребро между u и v . Гарантируется, что такое ребро есть.
3. $?$: Посчитать количество компонент связности в графе.

Вершины пронумерованы целыми числами от 1 до N . Во всех запросах $u \neq v$.

Формат выходных данных

Для каждого запроса типа ‘?’, Выведите количество компонент связности в момент запроса.

Пример

стандартный ввод	стандартный вывод
5 11	5
?	1
+ 1 2	1
+ 2 3	2
+ 3 4	
+ 4 5	
+ 5 1	
?	
- 2 3	
?	
- 4 5	
?	

Задача Н. Дерево Андрея

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Магистр Андрей очень любит деревья[†], поэтому у него есть дерево, состоящее из n вершин.

Но не все так просто. Магистр Тимофей решил украсть одну вершину из дерева. Если Тимофей украл вершину v из дерева, то вершина v и все ребра с концом в вершине v удаляются из дерева, при этом номера других вершин не меняются. Чтобы Андрей не расстраивался, Тимофей решил сделать получившийся граф снова деревом. Для этого он может добавлять ребра между произвольными вершинами a и b , однако при добавлении такого ребра он должен заплатить $|a - b|$ монет в Центр Помощи Магистрам.

Обратите внимание, что получившееся дерево **не содержит** вершину v .

Тимофей не определился, какую вершину v он удалит из дерева, поэтому он хочет знать для каждой вершины $1 \leq v \leq n$, какое минимальное количество монет нужно потратить, чтобы после удаления вершины v сделать граф снова деревом, а также какие ребра при этом нужно добавить.

[†]Деревом называется неориентированный связный граф без циклов.

Формат входных данных

Каждый тест состоит из нескольких наборов входных данных. Первая строка содержит одно целое число t ($1 \leq t \leq 10^4$) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит одно целое число n ($5 \leq n \leq 2 \cdot 10^5$) — количество вершин в дереве Андрея.

В следующих $n - 1$ строках содержится описание ребер дерева. i -я из этих строк содержит два целых числа u_i и v_i ($1 \leq u_i, v_i \leq n$) — номера вершин, соединённых i -м ребром.

Гарантируется, что данные рёбра образуют дерево.

Гарантируется, что сумма n по всем наборам входных данных не превосходит $2 \cdot 10^5$.

Формат выходных данных

Для каждого набора входных данных выведите ответ в следующем формате:

Для каждой вершины v (в порядке от 1 до n) в первой строке выведите два целых числа w и m — минимальное количество монет, которое нужно потратить, чтобы граф после удаления вершины v снова стал деревом, и количество добавленных ребер.

Далее выведите m строк, каждая из которых содержит два целых числа a и b ($1 \leq a, b \leq n, a \neq v, b \neq v, a \neq b$) — концы добавленного ребра.

Если существует несколько способов добавить ребра, вы можете вывести любое решение с минимальной стоимостью.

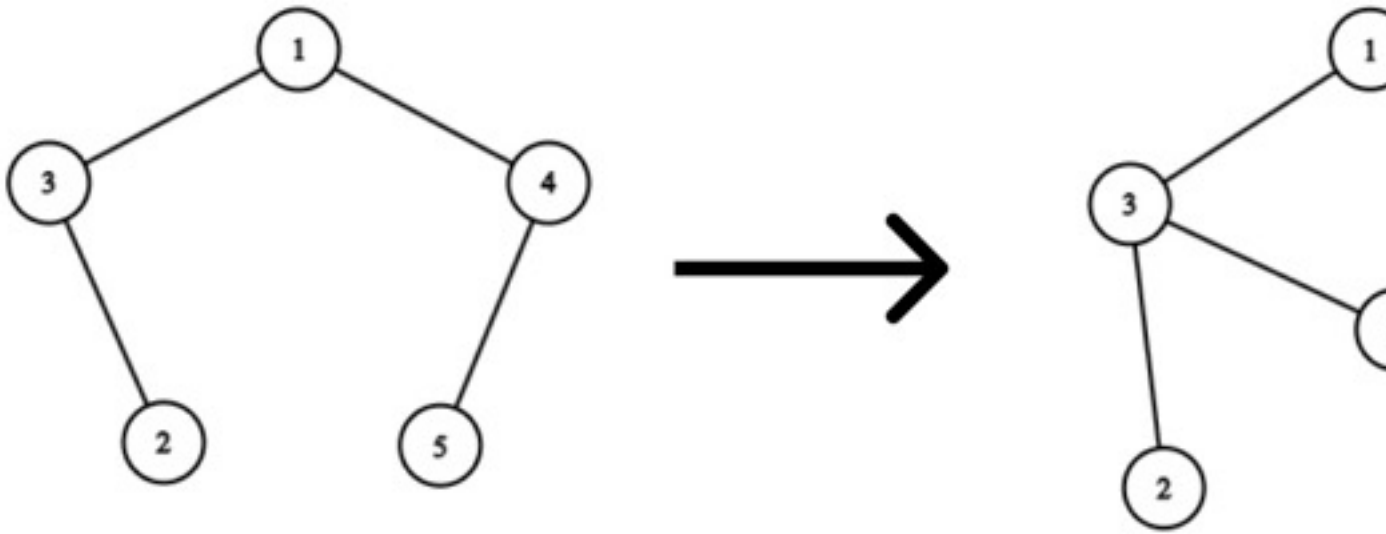
Пример

стандартный ввод	стандартный вывод
3	1 1
5	3 4
1 3	
1 4	0 0
4 5	
3 2	1 1
5	1 2
4 2	
4 3	2 1
3 5	3 5
5 1	
5	0 0
2 1	
1 5	0 0
1 4	
1 3	0 0
	1 1
	1 2
	1 1
	1 2
	1 1
	1 2
	3 3
	2 3
	4 5
	3 4
	0 0
	0 0
	0 0
	0 0

Замечание

В первом наборе входных данных:

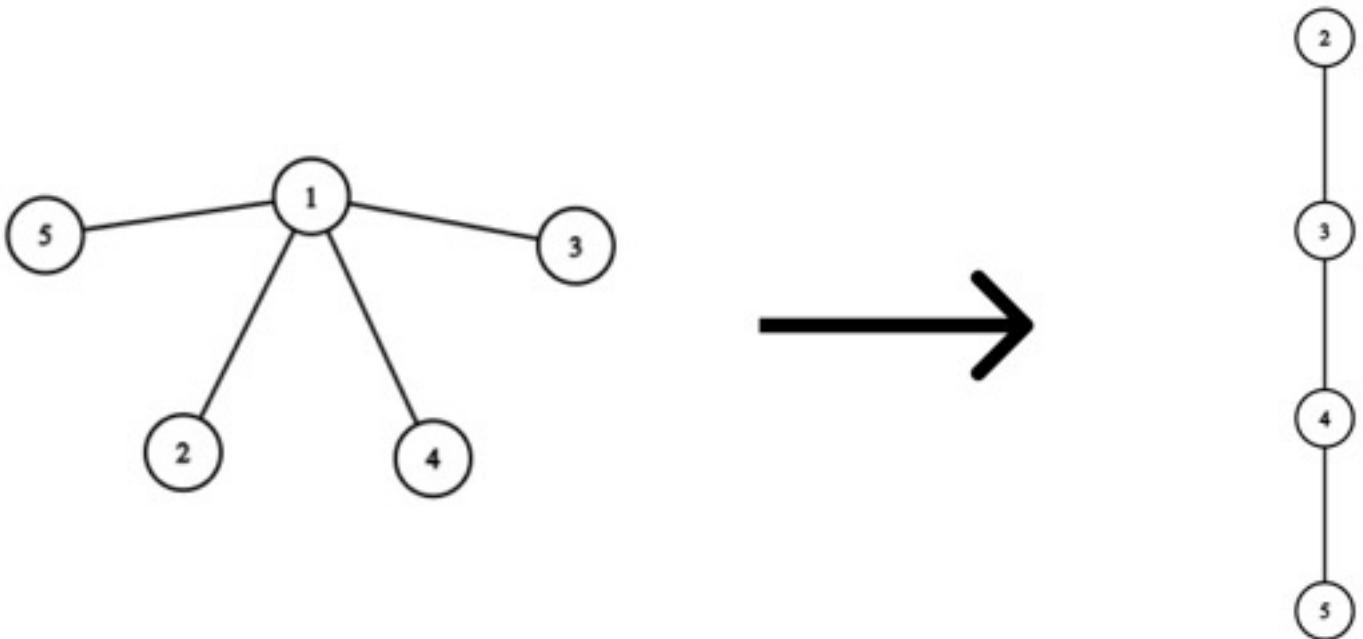
Рассмотрим удаление вершины 4:



Оптимальным решением будет провести ребро из вершины 5 в вершину 3. Тогда мы потратим $|5 - 3| = 2$ монеты.

В третьем наборе входных данных:

Рассмотрим удаление вершины 1:



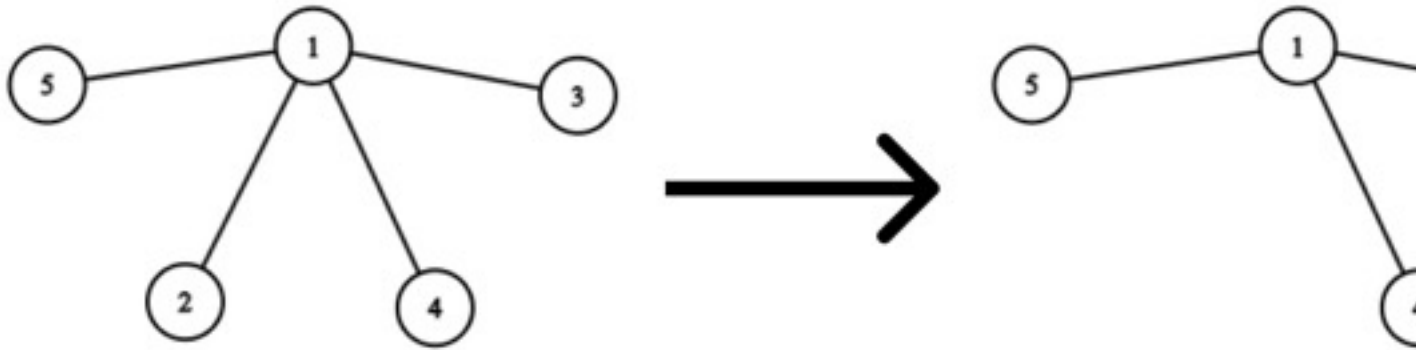
Оптимальным решением будет:

- Провести ребро из вершины 2 в вершину 3, потратив $|2 - 3| = 1$ монету.
- Провести ребро из вершины 3 в вершину 4, потратив $|3 - 4| = 1$ монету.

- Провести ребро из вершины 4 в вершину 5, потратив $|4 - 5| = 1$ монету.

Тогда сумарно мы потратим $1 + 1 + 1 = 3$ монеты.

Рассмотрим удаление вершины 2:



Ребра проводить не нужно, так как после удаления вершины граф останется деревом.