Задача 1. Alp bus

Входной файл	`aplusb.in` или ввод с клавиатуры
Выходной файл	`aplusb.out` или вывод на экран
Ограничение по времени	1 сек
Ограничение по памяти	256 МиБ
Максимальный балл за задачу	300

Напишите программу, которая складывает два числа.

Формат входных данных

Во входном файле находятся два числа, A и B. Числа целые и не превосходят по модулю 1000.

Формат выходных данных

Выведите в выходной файл одно число — сумму двух данных чисел.

Пример

Входной файл	*Выходной файл*
2 2	4
-4 2	-2

Задача 2. Нейросеть

Входной файл	`dl.in` или ввод с клавиатуры
Выходной файл	`dl.out` или вывод на экран
Ограничение по времени	1 сек
Ограничение по памяти	256 МиБ
Максимальный балл за задачу	100

Одним холодным зимним утром студент Вася решил освоить модное нынче глубокое обучение. Быстро нагуглив какой-то туториал, Вася погрузился в чтение. Пропустив скучную теорию, Вася сразу перешёл к примеру кода, и уже через полчаса нейросеть обучалась на картинках с котами, а старая видеокарта Васиного компьютера гудела, пытаясь справиться с вычислениями.

Спустя ещё час Вася получил обученную модель, которая делала всё что угодно, но только не то, что от неё ожидалось. Проанализировав ситуацию, Вася быстро понял причину — его нейросеть недостаточно крутая. Надо срочно её исправить!

В данной задаче нейросетью мы будем считать набор нейронов, разбитый на n слоёв, на i-м из которых находится $a_i > 0$ нейронов, и между всеми парами нейронов из соседних слоёв и только между ними есть нейронная связь (для лучшего понимания см. рисунок к первому примеру). Путём в нейросети назовём такую последовательность нейронов, в которой любые два соседних нейрона имеют нейронную связь. Крутостью нейросети Вася считает количество путей, состоящих из n нейронов, из которых первый нейрон находится в первом слое, а последний — в последнем (как нетрудно заметить, это количество равно $a_1 \cdot a_2 \cdot \ldots \cdot a_n$). Вася считает, что его нейросеть будет достаточно крутой, если её крутость будет не меньше некоторого числа K. При этом для более быстрого обучения желательно иметь как можно меньше нейронов. Помогите Васе построить такую нейросеть.

Формат входных данных

В единственной строке входных данных даны два целых числа n ($2 \leqslant n \leqslant 10^5$) и K ($1 \leqslant K \leqslant 10^{18}$) — число слоёв сети и минимальная крутость соответственно.

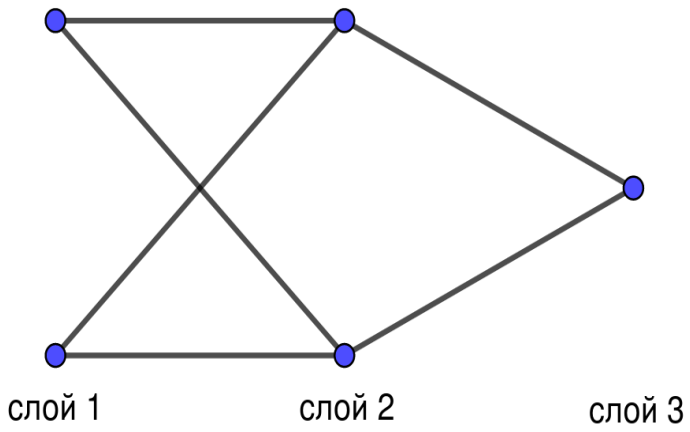
Формат выходных данных

Выведите числа $a_1, \ldots, a_n$ — размеры слоёв в сети с минимальным количеством нейронов, крутость которой не меньше K. Если правильных ответов несколько, выведите любой.

Пример

Входной файл	*Выходной файл*
3 4	2 2 1
10 1	1 1 1 1 1 1 1 1 1 1

Примечание: Нейросеть из первого примера



Задача 3. Посылки в заморозку

Входной файл	`freeze.in` или ввод с клавиатуры
Выходной файл	`freeze.out` или вывод на экран
Ограничение по времени	1 сек
Ограничение по памяти	256 МиБ
Максимальный балл за задачу	100

Всеберляндская командная олимпиада школьников по программированию (ВКОШП) проходит по стандартным правилам чемпионата мира по программированию: n командам дается 5 часов на решение m задач, и по традиции в последний час не показываются результаты попыток команд, видно только, какая команда посылала какие задачи.

Команды в итоговой таблице сортируются по количеству решенных задач (чем больше, тем лучше), а при равном количестве — по штрафному времени (чем меньше, тем лучше). Формально, место, занимаемое командой, определяется следующим образом:

(количество команд, у которых больше решенных задач) +
+(количество команд, у которых такое же число решенных задач, но строго меньше штрафное время)+1.

При равенстве числа задач и штрафного времени команды делят занимаемое ими место (в частности, несколько команд могут одновременно занимать первое место). Штрафное время определяется как сумма времен первых успешных попыток по всем сданным задачам плюс 20 минут за каждую неудачную попытку по каждой успешно сданной задаче. Обратите внимание, если команда уже успешно сдала задачу, все последующие посылки не влияют на штрафное время, а также что штрафное время за несданные задачи не начисляется.

Никита не любит интригу, а поэтому во время заморозки постоянно пытается угадать, может ли та или иная команда быть лидером. При этом он предполагает, что если команда послала некоторую задачу в заморозку, то все предыдущие посылки в заморозку по этой задаче были неверные. Однако, команды могут перепосылать сданные до заморозки задачи. Исходя из этого, вам нужно обрабатывать несколько запросы двух видов:

- 1 s i t — команда с названием s послала задачу i во время t.
- 2 s — Никиту интересует, может ли команда с названием s в данный момент быть на первом месте.

Формат входных данных

В первой строке записаны два числа n и m ($1 \leqslant n \leqslant 400$, $8 \leqslant m \leqslant 15$) — количество команд и задач соответственно.

Далее следует n описаний результатов команд. Каждой описание начинается с непустой строки s длиной не более 100 символов — названия команды. Название может содержать в себе заглавные и строчные буквы латинского алфавита, цифры, а также символы '`-`', '`#`' и '`_`'. Все названия различны.

Далее в m строках содержатся результаты команды по каждой задаче по итогам первых четырех часов. В начале записан вердикт по текущей задаче: '`.`' — команда не посылала данную задачу, '`+`' — команда сдала эту задачу, '`-`' — команда сделала несколько неудачных посылок, но не сдала задачу. Если вердикт не равен '`.`', то далее следуют два числа k и t ($0 \leqslant k \leqslant 100$, $0 \leqslant t \leqslant 239$) — количество неудачных попыток и время последней неудачной или первой удачной (если команда сдала задачу) попытки.

Описания команд разделяются пустой строкой.

После всех описаний записано целое число q ($1 \leqslant q \leqslant 2000$) — количество запросов. Далее в q строках записаны запросы согласно формату выше. Гарантируется, что для каждого запроса первого типа $240 \leqslant t \leqslant 299$. Гарантируется, что все запросы упорядочены по времени. Задачи в запросах первого типа нумеруются от 1 до m. Также гарантируется, что в запросах встречаются только команды, описанные выше.

Формат выходных данных

Для каждого запроса второго типа в отдельной строке выведите '`YES`', если данная команда может быть на первом месте в соответствующий момент, и '`NO`' в противном случае.

Пример

Входной файл	*Выходной файл*
3 8 lol . + 0 12 + 3 29 . - 4 201 - 1 100 + 1 56 + 2 228 kek . + 0 23 + 0 165 - 36 200 . . + 4 55 . cheburek - 5 239 + 1 5 + 0 30 . + 2 229 . + 10 45 + 20 203 8 1 lol 1 240 2 lol 1 kek 1 256 2 kek 1 kek 4 278 2 kek 1 cheburek 2 298 2 cheburek	YES NO NO YES

Примечание: Изначальные результаты команд: ‘lol’ — 4 задачи, 445 минут штрафа; ‘kek’ — 3 задачи, 323 минуты штрафа; ‘cheburek’ — 5 задач, 1172 минуты штрафа.

Если посылка команды ‘lol’ пройдет, то у нее станет 5 задач и 685 минут штрафа и она будет на первом месте.

Даже если пройдут обе посылки команды ‘kek’, у нее будет 5 задач и 1577 минут штрафа, и она не сможет подняться на первое место.

Задача 4. Счастливые дни

Входной файл	`happy.in` или ввод с клавиатуры
Выходной файл	`happy.out` или вывод на экран
Ограничение по времени	1 сек
Ограничение по памяти	256 МиБ
Максимальный балл за задачу	100

Широко известна легенда про ханойские башни — про монахов в ханойском храме, перекладывающих диски по определенным правилам до конца света.

Намного менее известна другая легенда — что в этом же храме есть N стержней, занумерованных от 1 до N. Рядом с каждым стержнем написано одно число от 1 до N, все числа различны. На каждом стержне лежит один золотой диск. Все диски разных размеров, и при сотворении мира диски лежали по порядку: самый маленький диск на первом стержне, и т.д., самый большой — на последнем.

Каждую ночь, в полночь, монахи проводят следующую операцию. Они снимают все диски со стержней, после чего перекладывают их в соответствии с числами, написанными около стержней: диск, лежавший на первом стержне, перекладывается на стержень с номером, равным числу, написанному около первого стержня; диск, лежавший на втором стержне — на стержень с номером, равным числу, написанному около второго стержня, и так далее. Формально: если рядом с i-м стержнем написано число a_i, то диск, лежавший на i-м стержне, перекладывается на стержень с номером a_i.

Получившееся расположение дисков определяет счастливость наступившего дня. Счастливость будет равна количеству пар дисков a, b, лежащих в «неправильном порядке», т.е. таких, что диск a больше диска b, но лежит на стержне с меньшим номером.

Напишите программу, которая по порядковому номеру дня с сотворения мира определяет его счастливость.

Формат входных данных

В первой строке записано натуральное число n ($1 \leqslant n \leqslant 100\,000$) — количество стержней и дисков.

Во второй строке записано n натуральных чисел a_i ($1 \leqslant a_i \leqslant n$) — числа, написанные рядом со стержнями. Гарантируется, что все эти числа различны.

В третьей строке записано одно неотрицательное целое число S — номер дня с сотворения мира. День, когда был сотворен мир, считается днем номер ноль. Гарантируется, что число S не превосходит $10^{100\,000}$ (т.е. единицы с 100 000 нулями).

Формат выходных данных

В единственной строке выведите счастливость S-го дня.

Система оценивания

Решения, работающие для $n \cdot S \leqslant 10^8$, будут набирать не менее 30 баллов.

Решения, работающие для $n \cdot S \leqslant 10^8$ и $n \leqslant 10^4$, будут набирать не менее 20 баллов.

Пример

Входной файл	*Выходной файл*
4 1 3 2 4 3	1

Примечание: Занумеруем диски в примере по порядку от меньшего к большему.

В день сотворения мира диски лежали в следующем порядке: $1, 2, 3, 4$.

На следующий день (день номер 1) диски лежали в следующем порядке: $1, 3, 2, 4$.

Во второй день — $1, 2, 3, 4$.

В третий день — $1, 3, 2, 4$.

В третий день есть только одна пара дисков, лежащих в неправильном порядке — диски 3 и 2, — поэтому счастливость третьего дня равна 1.

Задача 5. День города

Входной файл	`path.in` или ввод с клавиатуры
Выходной файл	`path.out` или вывод на экран
Ограничение по времени	4 сек
Ограничение по памяти	256 МиБ
Максимальный балл за задачу	100

Город Н широко известен своей системой дорог: в городе есть n перекрёстков и m широких асфальтированных дорог, каждая из которых соединяет два перекрёстка. Пару десятков лет назад для решения проблем с пробками все дороги сделали платными: проезд по i-й дороге в любую сторону стоит c_i монет. Пробки с тех пор никуда не исчезли, а цены так и остались.

К очередному дню города мэр города Н решил в качестве подарка жителям выбрать перекрёсток и сделать бесплатными все дороги, соединяющие его с другими перекрёстками. Однако не всё так просто. Дело в том, что каждый день мэр ездит из своего дома на 1-м перекрёстке в мэрию на n-м перекрёстке, а проезд стоит денег. Поэтому мэр хочет выбрать перекрёсток и сделать соседние с ним дороги бесплатными так, чтобы в результате тратить как можно меньше денег на дорогу на работу (т.е. чтобы в итоге оптимальный путь от перекрестка 1 до перекрестка n был бы как можно дешевле).

Формат входных данных

В первой строке входных данных даны два целых числа n ($2 \leqslant n \leqslant 4 \cdot 10^5$) и m ($1 \leqslant m \leqslant 4 \cdot 10^5$) — число перекрёстков и дорог соответственно.

В каждой из следующих m строк идут три целых числа $v,\ u,\ c$ ($1 \leqslant v, u \leqslant n,\ 1 \leqslant c \leqslant 10^9$) — номера перекрёстков, соединяемых очередной дорогой, и стоимость проезда по этой дороге. Гарантируется, что никакие две дороги не соединяют одну и ту же пару перекрёстков, и ни одна дорога не соединяет какой-либо перекрёсток с ним же.

По каждой дороге можно ездить в любую сторону. Гарантируется, что по дорогам можно проехать от перекрёстка 1 до перекрёстка n.

Формат выходных данных

Выведите одно целое число — минимальную стоимость проезда от перекрёстка 1 до перекрёстка n при оптимальном выборе перекрёстка, смежные с которым дороги будут бесплатны.

Система оценивания

В тестах общей стоимостью не менее 40 баллов будет выполняться условие $n, m \leqslant 1000$.

Пример

Входной файл	*Выходной файл*
3 2 1 2 1 2 3 2	0
5 5 1 2 3 1 3 4 2 3 1 3 4 5 4 5 6	4

Примечание: В первом примере оптимально обнулить дороги, смежные с перекрёстком 2.

Во втором примере оптимально обнулить дороги, смежные с перекрёстком 4.

Задача 6. Преобразование выражения

Входной файл	`sum.in` или ввод с клавиатуры
Выходной файл	`sum.out` или вывод на экран
Ограничение по времени	1 сек
Ограничение по памяти	256 МиБ
Максимальный балл за задачу	100

Вася обучает свою младшую сестру Машу сложению целых чисел. Вася уже написал некоторое корректное выражение и хочет предложить Маше посчитать значение этого выражения, то есть найти сумму записанных чисел. Выражение считается корректным, если выполнены следующие условия:

- выражение содержит одно или несколько целых неотрицательных чисел, разделенных знаком '+' (без кавычек);
- числа не содержат ведущих нулей (однако, числа, равные 0, допустимы);
- каждый знак '+' находится между двумя числами.

Когда всё выражение уже было готово, Вася вспомнил, что Маша может не знать достаточно больших чисел, чтобы найти сумму. Васе известно, что Маша знает целые числа от 0 до N включительно, соответственно, и сумма чисел в выражении должна не превосходить N. У Васи еще есть время, чтобы поменять не более K символов в выражении (можно заменять любой символ на цифру или знак '+'), при этом выражение должно остаться корректным, а сумма получившихся чисел должна быть не более N. Кроме того, Вася хочет упростить задачу для сестры, поэтому он будет пытаться сделать так, чтобы слагаемые в выражении были как можно меньше, а именно — максимальное слагаемое в выражении должно быть минимально возможным. Помогите найти Васи оптимальное выражение, удовлетворяющее этим условиям.

Формат входных данных

Первая строка содержит три целых числа — L, N, K — количество символов в исходном выражении, максимальное число, которое знает Маша, и максимально допустимое количество замен соответственно ($1 \leqslant L \leqslant 300$, $0 \leqslant N \leqslant 10^9$, $0 \leqslant K \leqslant L$). Во второй строке содержится исходное выражение, записанное Васей. Гарантируется, что выражение является корректным и содержит ровно L символов (цифр и знаков '+').

Формат выходных данных

Если не существует преобразования выражения, удовлетворяющего всем условиям, нужно вывести `-1` (см. пример 2). Если допустимые преобразования существуют, нужно вывести преобразованное выражение, в котором максимальное слагаемое будет как можно меньше. Выведенное выражение должно быть корректным.

Если существует несколько таких выражений, можно вывести любое из них (см. пример 3). Обратите внимание, что Васе не обязательно использовать все K замен (см. пример 4).

Система оценивания

Тесты, для которых выполняется $L \leqslant 16$, в сумме оцениваются в 40 баллов.

Пример

Входной файл	*Выходной файл*
8 60 1 123+45+6	1+3+45+6
5 20 1 21+12	-1
5 10 2 4+4+4	0+0+4
5 100 2 4+4+4	4+4+4

Задача 7. Контроль светофоров

Входной файл	`trfl.in` или ввод с клавиатуры
Выходной файл	`trfl.out` или вывод на экран
Ограничение по времени	1 сек
Ограничение по памяти	256 МиБ
Максимальный балл за задачу	100

Иван Дмитриевич работает монтажником-контролером в городских электросетях. Его работа состоит в том, чтобы проверять установку новых светофоров.

Светофоры, как известно, имеют три лампы: красную, желтую и зеленую. От каждой лампы идет провод к специальному контрольному реле; с другой стороны, к этому же реле идут три провода от компьютера, управляющего светофором. Каждый из этих проводов соответствует своему цвету лампы; по каждому проводу компьютер может посылать, а может и не посылать управляющий сигнал. В зависимости от наличия или отсутствия сигнала на входных проводах реле включает или выключает соответствующие лампы.

А именно, бывают два типа реле: реле первого типа держит лампу включенной, пока по соответствующему проводу *подается* управляющий сигнал, и наоборот: пока по проводу управляющий сигнал не подается, лампа не горит. Реле второго типа держит лампу включенной, если по соответствующему проводу управляющий сигнал *не подается*, и наоборот.

Например, с реле второго типа, чтобы на светофоре горели красный и желтый сигналы, а зеленый не горел, на реле надо подать управляющий сигнал только по «зеленому» проводу, а по «красному» и «желтому» не подавать.

Тип используемого реле вносится в управляющую программу светофора, и типичной ошибкой монтажников является указание неверного типа реле. В такой ситуации, если компьютер хочет, чтобы на светофоре горела некоторая лампа, она не будет гореть, и наоборот.

Другой типичной ошибкой является подключить лампы в обратном порядке, т.е. перепутать провода, отвечающие зеленой и красной лампе. В таком случае при попытке включить красную лампу загорается зеленая, и наоборот. Желтую лампу невозможно перепутать с другими лампами, т.к. она находится посередине.

Нередко монтажники допускают и обе ошибки вместе, в таком случае, например, при попытке включить красную лампу на самом деле выключается зеленая лампа, и т.д.

Иван Дмитриевич прибыл на очередной только что смонтированный светофор. Он попытался через программу включить некоторые лампы, оставив другие выключенными. По тому, какие лампы на самом деле включились, определите, какие ошибки допустили монтажники.

Формат входных данных

Во входных данных находятся несколько тестовых примеров. Соответственно, на первой строке входных данных записано одно число T — количество тестовых примеров ($1 \leqslant T \leqslant 3$).

Далее следуют $2T$ строк, описывающих тестовые примеры. Каждый пример описывается двумя строками. На первой строке записаны три символа '`0`' или '`1`' (без пробелов), соответственно указывающие, что Иван Дмитриевич хотел сделать с красной, желтой и зеленой лампами. '`0`' обозначает, что он хотел её выключить, '`1`' – включить. На второй строке в том же формате записано, что на самом деле получилось: '`0`' обозначает, что лампа не включилась, '`1`' – включилась.

Формат выходных данных

Выведите T строк. А именно, для каждого примера:
выведите '`OK`', если монтажники точно ничего не напутали,
выведите '`1`', если монтажники неправильно указали реле, но точно подключили провода правильно,
выведите '`2`', если монтажники перепутали порядок проводов, но точно не перепутали реле,
выведите '`3`', если монтажники и перепутали порядок проводов, и перепутали реле,
выведите '`?`', если наблюдаемую картину можно объяснить несколькими способами,
выведите '`ERROR`', если монтажники или кто-то еще, видимо, допустили какую-то еще ошибку.

Пример

Входной файл	*Выходной файл*
2 001 001 001 011	OK 3