Problem C. Коммивояжёр

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	1 секунда
Memory limit:	256 мебибайт

В Байтландии N городов. Некоторые пары городов соединены дорогами с односторонним движением так, что любая пара городов соединена напрямую не более, чем одной дорогой. Для каждой дороги определена стоимость проезда по ней.

Коммивояжёр собирается посетить минимум по одному разу все города и вернуться в город, из которого он выехал. При этом он может посещать города проездом (то есть если план коммивояжёра выглядит как 1, 0, 2, то маршрут **1**, 2, **0**, **2**, **1** его устраивает (жирным выделена соответствующая плану последовательность), то маршрут 1, 2, 0, 1 не устраивает.

Требуется найти минимальную сумму, за которую он сможет реализовать план, или определить, что все города объехать невозможно.

Input

Первая строка входных данных содержит одно целое число N — количество городов в Байтландии ($1 \le N \le 200$).

Во второй строке заданы N различных целых чисел от 0 до $N - 1$ — план поездки, составленный коммивояжёром. Обратите внимание, что маршрут должен быть замкнутым, то есть коммивояжёр должен вернуться в город, указанный в маршруте первым.

Далее следуют N строк, каждая из которых содержит N целых чисел от -1 до 10^4. d_{ij} — j-е число в i-й строке — задаёт стоимость проезда из i-го города в j-й или -1, если из города i в город j нет прямой дороги. Гарантируется, что $d_{ii} = 0$.

Output

Выведите одно число — наименьшую сумму, потратив которую, коммивояжёр сможет реализовать свой план, или -1, если объехать все города и вернуться в стартовый невозможно.

Examples

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

Problem D. Овощной суп

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	1 секунда
Memory limit:	256 мебибайт

Чтобы приготовить овощной суп, овощи надо нарезать как можно более одинаковыми кусочками. У Вас есть N овощей разного размера. Требуется за минимальное число разрезов нарезать овощи так, чтобы отношение массы самого мелкого куска к самому крупному превосходило заданную константу T. За один разрез можно разрезать один кусок на два куска меньшего размера.

Input

Первая строка входных данных содержит два числа — вещественное число T ($0.5 < T < 1$) и целое число N ($1 \le N \le 1000$) — требуемое отношение и количество овощей соответственно.

Вторая строка содержит N целых чисел w_i — вес i-го из овощей ($1 \le w_i < 10^6$).

Output

Выведите минимальное число разрезов, требуемое для того, чтобы отношение весов минимального и максимального куска овощей было больше T. Гарантируется, что ответ меньше 500 и что при изменении T на 10^{-4} ответ не изменится.

Example

стандартный ввод	стандартный вывод
0.99 3 2000 3000 4000	6
0.80 2 1000 1400	3

Problem E. Обмен телами

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	1 секунда
Memory limit:	256 мебибайт

Для путешествий по другим галактикам жителями третьей планеты системы Тау Кита был изобретён прибор, позволяющий обмениваться телами — то есть путешественник A оказывается в теле путешественника B, а путешественник B — в теле путешественника A. Прибор позволяет делать неограниченное количество обменов при условии, что пары тел, участвующих в обмене, не повторяются (это сделано для того, чтобы удвоить количество продаваемых приборов).

Алиса и Боб решили подзаработать и предложили свои услуги для того, чтобы обойти это ограничение. Пусть есть группа из N путешественников, которая с помощью одного прибора провела некоторое количество обменов и которой нужно вернуться в исходное состояние (каждый участник группы в своём теле). Алиса и Боб предлагают цепочку обменов со своим участием, которая решает задачу без покупки нового прибора (при этом в итоге Алиса и Боб также оказываются в своих телах).

Ваша задача — написать программу, которая помогает Алисе и Бобу составить план такой цепочки обменов.

Input

Входные данные состоят из не более 15 тестовых примеров.

Каждый тестовый пример состоит из двух строк. Первая строка содержит два целых числа N и M ($1 \le N, M \le 10^5$) — количество путешественников в группе и количество уже произведённых обменов, соответственно. Путешественники занумерованы последовательными целыми числами от 0 до $N-1$, Алиса и Боб получают номера N и $N+1$. Вторая строка содержит $2M$ чисел и имеет вид $a_1\ b_1\ a_2\ b_2\ \ldots\ a_m\ b_m$, где a_i и b_i ($0 \le a_i, b_i \le N-1$) — пары тел, участвовавших в i-м обмене. Гарантируется, что в одном тестовом примере ни одна пара тел не встречается дважды.

Вывод завершается тестовым примером с $N = 0$, обрабатывать который не требуется.

Output

Для каждого тестового примера выведите одну строку, содержащую последовательность обменов, возвращающих каждого участника в своё тело. Формат последовательности аналогичен второй строке формата ввода. Количество обменов не должно превышать $5 \cdot 10^6$. Если обменов не требуется, выведите пустую строку. Последовательность не должна быть кратчайшей; если ответов несколько, выведите любой.

Example

стандартный ввод	стандартный вывод
4 3 0 1 2 3 0 2 4 6 0 1 2 3 0 2 1 3 0 3 1 2 100000 0 100000 1 0 1 100000 1 0 1 0	1 3 0 3 1 2 0 1 2 3 0 2 1 3 0 3 1 2 2 3 0 2 1 3 0 3 1 2 100001 3 0 100001 1 3 0 3 1 100001

Problem F. Раздел строки

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	1 секунда
Memory limit:	256 мебибайт

Пусть S — строка длины $2N$, а каждому символу в строке S соответствует целое число как его вес. Вес подпоследовательности T строки S (обозначаемый как $weight(T)$), определяется как сумма весов всех входящих в неё символов. Ваша задача — разбить S на две подпоследовательности T_1 и T_2, каждая из которых имеет длину N так, что T_1 равна T_2 и значение $|weight(T_1) - weight(T_2)|$ минимально.

Input

Первая строка входа содержит целое число N ($1 \le N \le 20$). Вторая строка содержит строку S длины $2N$. Каждый символ в этой строке или '`a`', или '`b`'. Третья строка содержит $2N$ целых положительных чисел w_i ($1 \le w_i \le 10^6$), где i-е число — вес i-го символа в строке S.

Output

Выведите наименьшее значение $|weight(T_1) - weight(T_2)|$. Если нельзя разбить строку S на две одинаковые подпоследовательности, выведите -1.

Examples

стандартный ввод	стандартный вывод
2 abab 3 1 10 5	11
3 aaabbb 1 1 1 2 2 2	-1
3 abaaba 4 6 10 5 3 4	2

Problem G. Облачные вычисления

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	1 секунда
Memory limit:	256 мебибайт

Рассмотрим одну из проблем, связанных с облачными вычислениями. Например, если вы отправляете на сервер две команды $op1$ и $op2$, ожидая, что команда $op1$ выполнится первой, а после неё выполнится $op2$. В связи со сбоями сети, $op1$ может прибыть на сервер раньше, чем $op2$.

Чтобы сервер мог восстановить порядок, в котором команды должны выполняться, у каждой команды есть атрибут — время отправки. Таким образом, если сервер получает две команды в формате $(op1, t_1)$ и $(op2, t_2)$, где $t_1 < t_2$, то в случае, если команда $(op2, t_2)$ пришла первой, она выполнится сразу же. А после того, как придёт команда $(op1, t_1)$, сервер установит, что она должна была быть выполнена перед $op2$ (так как $t_1 < t_2$), таким образом, сервер откатит $op2$, выполнит $op1$ и уже после этого снова выполнит $op2$.

В этой задаче вы должны проэмулировать вышеуказанный процесс. Сервер будет моделироваться стеком со следующей системой команд:

- `push x t` – положить x на стек (t — время отправки);
- `pop t` – взять верхний элемент стека (t — время отправки);
- `peak t` – напечатать верхний элемент стека (t — время отправки).

Когда команда op со временем отправки t прибывает на сервер, сервер делает следующие действия:

Step 1: откатить все команды "`push`" и "`pop`", у которых время больше t.

Step 2: выполнить op.

Step 3: снова применить все "`push`" и "`pop`", которые откатывались на первом шаге.

Сервер не должен откатывать команды "`peak`". Иначе говоря, каждая команда "`peak`" выполняется ровно один раз после того, как она поступает на сервер.

По последовательности команд, прибывающих на сервер, просимулируйте работу процессора. Изначально стек пуст. Кроме того:

1. Все команды "`pop`" корректны (то есть не бывает ситуации, когда перед выполнением этой команды стек пуст)
2. Все времена отправки команд различны.

Input

Входные данные начинаются строкой, содержащей целое число N ($1 \le N \le 5 \cdot 10^4$) — количество команд. Последующие N строк задают команды, по одной на строку, в форматах "`push x t`", "`pop t`", "`peak t`", где $0 \le x, t \le 10^9$.

Команды даются в порядке прибытия на сервер.

Output

Для каждой операции "`peak`" выведите результат её выполнения. Если стек к моменту вызова "`peak`" пуст, выведите -1.

Examples

стандартный ввод	стандартный вывод
7 push 100 3 push 200 7 peak 4 push 50 2 pop 5 peak 6 peak 8	100 50 200
4 push 25 1 pop 5 peak 6 peak 3	-1 25
4 push 10 1 peak 7 pop 3 peak 4	10 -1

Problem H. Ипохондрия

Input file:	*стандартный ввод*
Output file:	*стандартный вывод*
Time limit:	2 секунда
Memory limit:	256 мебибайт

Удовлетворённость пациента жизнью за день оценивается величиной x_t по шкале от -10^9 до 10^9. Величину $s_i = |x_{i-1} - x_i|$ будем называть *вариацией настроения* дня i.

Будем говорить, что пациент находится в состоянии ипохондрии, прогрессирующей со дня k по день m, если для любого номера дня i, $k < i < m$ значения вариации возрастают ($s_i < s_{i+1}$). И если $x_i < x_k$, то $x_{i+1} > x_k$ или если $x_i > x_k$, то $x_{i+1} < x_k$.

Пациент находится в состоянии ипохондрии, затухающей со дня k по день m, если для любого номера дня i $k \le i < m - 1$ значения вариации убывают ($s_i > s_{i+1}$), и если $x_i < x_m$, то $x_{i+1} > x_m$ или если $x_i > x_m$, то $x_{i+1} < x_m$.

Например, последовательность 5, 6, 4, 7, 1, 9, 6 — ипохондрия, прогрессирующая с 1-го по 6-й дни, а 6, 9, 1, 7, 4, 6, 5 — ипохондрия, затухающая со 2-го по 6-й дни.

Имеется статистика удовлетворённости жизнью за период N дней. Необходимо выбрать наибольшую по продолжительности последовательность дней, не обязательно идущих подряд (можно с пропусками), соответствующую ипохондрии затухающей, и, соответственно, ипохондрии прогрессирующей.

Input

В первой строке входных данных записано целое число N ($1 \le N \le 2000$). Во второй строке записаны N различных целых чисел x_t, разделённых пробелом ($-10^9 \le x_t \le 10^9$).

Output

Выведите два числа, разделённых пробелом — наибольшую длину подпоследовательности, соответствующей ипохондрии затухающей и ипохондрии прогрессирующей, соответственно.

Example

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