Задача A. Весы

Имя входного файла:	`scales.in`
Имя выходного файла:	`scales.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Ежи всегда носил с собой какую-то вещь. Определённую...
Андрей Андрианов, “Ежи и Петруччо”

Роботу Кузе поручили взвесить вещь. Для этого ему были предоставлены двухчашечные весы и набор из N гирь. Ему подготовили рабочее место — положили вещь и некоторые гири на одну чашу весов, а остальные гири — на другую.

Да только вот беда: Кузя — весьма несовершенный робот. Начнём с того, что в силу конструктивных особенностей и расположения весов он не может просто снять гирю с весов. Всё, что может сделать робот — это переложить определённую гирю с одной чаши весов на другую. Хорошо хоть, что он различает гири и запоминает, где какая находится.

Другая проблема заключается в том, что Кузя не может по положению весов определить, какая чаша на данный момент легче, а какая — тяжелее. Лишь когда чаши весов приходят в равновесие, раздаётся мелодичный звон, который робот может уловить. Логично утверждать, что если бы весы изначально находились в положениии равновесия, то Кузе бы не поручили эту работу.

Наконец, хотя Кузя и различает все гири, он не имеет понятия о том, каков их вес. Он не может даже определить, верно ли, что два набора гирь имеют одинаковый вес — для этого ему пришлось бы убрать вещь с весов, а этого он сделать не может.

Понятно, что Кузе очень не нравится такая работа, и он хочет покончить с ней как можно скорее. Для этого ему, очевидно, необходимо будет перебрать все положения гирь, которые он может получить из исходного. Если при переборе всех положений весы ни разу не придут в равновесие, значит, Кузя не может точно взвесить данную ему вещь. В противном случае он остановится, как только услышит звон весов.

Кузя хочет минимизировать количество перемещений гирь. Помогите ему составить план того, когда какую гирю перемещать, чтобы в худшем случае, если точное взвешивание невозможно, перебрать все положения гирь и доказать это.

Формат входного файла

В первой строке входного файла записано число N ($1 \leqslant N \leqslant 14$) — количество гирь. Во второй строке даны через пробел N чисел $A_1, A_2, \ldots, A_N$, каждое из которых равно 0 или 1. $A_i = 0$ означает, что гиря с номером i изначально находится на той же чаше весов, что и вещь, $A_i = 1$ — на противоположной.

Формат выходного файла

На первой строке выходного файла выведите одно число — количество перемещений гирь M ($0 \leqslant M \leqslant 1\,000\,000$) (помните, что M нужно минимизировать). На второй строке выведите M чисел B_j ($1 \leqslant B_j \leqslant N$) через пробел — какую гирю следует переложить на j-ом шаге с той чаши весов, на которой она сейчас находится, на противоположную.

Примеры

scales.in	scales.out
1 0	1 1
2 0 0	3 1 2 1
2 0 1	3 2 1 2

Задача B. Тест Миллера—Рабина

Имя входного файла:	`mr.in`
Имя выходного файла:	`mr.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Существует множество способов проверки числа на простоту. Например, если проверяемое число N достаточно мало, то можно просто поделить N на все простые числа, не превосходящие $\sqrt{N}$. Если N делится нацело хотя бы на одно из них — значит, оно составное, в противном же случае оно является простым.

Однако, когда число N велико, такой метод может потребовать от проверяющего слишком много времени — ведь трудоёмкость растёт экспоненциально от длины числа N. В настоящее время известно несколько способов определить простоту числа точно, но все они, как правило, работают слишком долго.

Поэтому чаще применяют способы, определяющие простоту числа с некоторой вероятностью. Один из наиболее быстрых и вместе с тем довольно надёжных способов известен как тест Миллера—Рабина. Ознакомимся с ним подробнее.

Сначала проверим, что N нечётно и больше, чем 1 (в противном случае проверка тривиальна). Представим $N-1$ как $2^s \cdot d$; заметим, что $s \geqslant 1$.

Теперь для нескольких различных $a \in [1, N-1]$ произведём следующую процедуру. Рассмотрим числа $k_r = a^{2^r \cdot d}$ для $r = 0, 1, \ldots, s-1$. Если $k_0 \mod N \neq 1$ и ни одно из k_r не совпадает с -1 по модулю N (другими словами, $k_r \mod N \neq N-1$), число N — составное. В противном случае мы повторяем эту процедуру для следующего a. Чем больше чисел a было проверено, тем больше вероятность того, что число N — простое. Обычно в качестве a подставляют первые несколько простых чисел — $2, 3, 5, 7, 11, \ldots$.

Мы не будем сейчас останавливаться на том, почему тест Миллера—Рабина работает. Наша задача заключается в другом — по числу N определить, каково же наименьшее простое число a, для которого описанная выше процедура приведёт к установлению того, что N — составное (разумеется, если это так). Число a не окажется слишком большим — известно, что наименьшее нечётное составное число N такое, что для него не срабатывают проверки с $a = 2, 3, 5, 7, 11$, равно 2152302898747.

Формат входного файла

В первой строке входного файла записано число N ($1 \leqslant N \leqslant 10^9$).

Формат выходного файла

Если N чётно или равно единице, и тест Миллера—Рабина неприменим, выведите -1. Если число N нечётное и простое, выведите 0. Иначе выведите наименьшее простое число a такое, что при его проверке по приведённому выше алгоритму выяснится, что N — составное.

Примеры

`mr.in`	`mr.out`
2	-1
15	2
4	-1
821	0
2047	3

Задача C. Такси

Имя входного файла:	`taxi.in`
Имя выходного файла:	`taxi.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Такси едет по прямому шоссе длины l. В начальный момент времени оно имеет скорость 1 и расположено в начале шоссе. Имея скорость v, такси за одну секунду изменяет свою координату на v, и после этого может выбрать скорость от $v - 2$ до $v + 1$ на следующую секунду; при этом в каждый момент времени $0 \leqslant v \leqslant 10$. Скорость всегда выражается целым числом.

В некоторых точках шоссе расположены светофоры. Если на светофоре горит красный свет, такси не может проехать, пока не загорится зелёный.

Водитель такси знает, когда переключаются все светофоры на шоссе, и хочет оказаться в его конце за минимальное время. Поскольку шоссе заканчивается железнодорожным переездом, конечная скорость v_{end} должна быть равна 1.

Выведите время в секундах, за которое такси проедет шоссе.

Формат входного файла

В первой строке входного файла записаны два числа l и n, разделённые пробелами ($1 \leqslant l \leqslant 500$, $0 \leqslant n < l$). В последующих n строках даны описания светофоров. Строка описания светофора имеет вид $x\, t_0\, t_{green}\, t_{red}$ и означает, что светофор имеет координату x, изначально на нём t_0 секунд горит красный, а затем каждые t_{green} секунд горит зелёный, а каждые последующие t_{red} секунд — красный. Известно, что $0 < x < l$, $0 < t_0, t_{green}, t_{red} \leqslant 20$. Никакие две координаты x светофоров не совпадают; светофоры даны в порядке возрастания координаты.

Формат выходного файла

Выведите единственное число — количество секунд, необходимых для того, чтобы оказаться в конце шоссе со скоростью 1 при заданных ограничениях.

Примеры

taxi.in	taxi.out
1 0	1
9 0	4
60 1 45 9 1 1	12
10 0	5
10 5 1 1 1 1 3 3 3 3 5 5 5 5 7 7 7 7 9 9 9 9	10