Задача Picasso. Зебры

Имя входного файла:	`input.txt` или стандартный поток ввода
Имя выходного файла:	`output.txt` или стандартный поток вывода
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Олег ведёт историю прожитых им дней, отмечая в ней каждый день либо как хороший, либо как плохой. Олег называет *зеброй* непустую последовательность дней, начинающуюся и заканчивающуюся плохим днём, в которой хорошие и плохие дни чередуются. Будем обозначать плохие дни за `0`, а хорошие за `1`. Тогда, например, последовательности дней `0`, `010`, `01010` являются зебрами, а последовательности `1`, `0110` и `0101` не являются.

Олег сообщил вам историю прожитых им дней в хронологическом порядке в виде строки из символов `0` и `1`, и теперь вас интересует, можно ли разбить историю Олега на несколько **подпоследовательностей**, каждая из которых является зеброй, и, если да, то как это можно сделать. Каждый день должен попасть ровно в одну подпоследовательность. Дни в каждой подпоследовательности должны следовать в хронологическом порядке. Обратите внимание, что подпоследовательность не обязана образовывать группу подряд идущих дней.

Формат входных данных

В единственной строке входных данных записана непустая строка s, состоящая из символов `0` и `1`, которая описывает историю дней, прожитых Олегом. Длина строки (обозначаемая дальше как $|s|$) не превосходит 200 000 символов.

Формат выходных данных

Если существует способ разбить историю дней на подпоследовательности, являющиеся зебрами, то в первой строке выведите число k ($1 \leqslant k \leqslant |s|$) — получившееся количество подпоследовательностей. В i-й из k последующих строк выведите сперва целое число l_i ($1 \leqslant l_i \leqslant |s|$) — длину i-й подпоследовательности, а затем l_i индексов, обозначающих номера дней в истории, составляющих подпоследовательность. Номера должны следовать в порядке возрастания, нумерация начинается с единицы. Каждый номер от 1 до n должен попасть ровно в одну подпоследовательность. Если разбиения на подпоследовательности-зебры не существует, то выведите `-1`.

Подпоследовательности можно выводить в любом порядке. Если возможных разбиений на подпоследовательности-зебры несколько, то разрешается вывести любое из них. Минимизировать или максимизировать величину k не требуется.

Примеры

ввод	вывод
0010100	3 3 1 3 4 3 2 5 6 1 7
111	-1

Система оценки

Тесты к этой задаче состоят из трёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов предыдущих групп.

Группа	Баллы	Дополнительные ограничения	Комментарий
		$\lvert s\rvert$	
0	0	–	Тесты из условия
1	31	$\lvert s\rvert \leqslant 10$	–
2	29	$\lvert s\rvert \leqslant 2000$	–
3	40	–	–

Задача Kandinsky. Обновление дата-центров

Имя входного файла:	`input.txt` или стандартный поток ввода
Имя выходного файла:	`output.txt` или стандартный поток вывода
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

У компании BigData Inc. есть n дата-центров, пронумерованных от 1 до n, расположенных по всему миру. В этих дата-центрах хранятся данные клиентов компании (как можно догадаться из названия — большие данные!)

Основой предлагаемых компанией BigData Inc. услуг является гарантия возможности работы с пользовательскими данными даже при условии выхода какого-либо из дата-центров компании из доступности. Подобная гарантия достигается путём использования *двойной репликации* данных. Двойная репликация — это подход, при котором любые данные хранятся в двух идентичных копиях в двух различных дата-центрах.

Про каждого из m клиентов компании известны номера двух различных дата-центров $c_{i,1}$ и $c_{i,2}$, в которых хранятся его данные.

Для поддержания работоспособности дата-центра и безопасности данных программное обеспечение каждого дата-центра требует регулярного обновления. Релизный цикл в компании BigData Inc. составляет один день, то есть новая версия программного обеспечения выкладывается на каждый компьютер дата-центра каждый день.

Обновление дата-центра, состоящего из множества компьютеров, является сложной и длительной задачей, поэтому для каждого дата-центра выделен временной интервал длиной в час, в течение которого компьютеры дата-центра обновляются и, как следствие, могут быть недоступны. Будем считать, что в сутках h часов. Таким образом, для каждого дата-центра зафиксировано целое число u_j ($0 \leqslant u_j \leqslant h-1$), обозначающее номер часа в сутках, в течение которого j-й дата-центр недоступен в связи с плановым обновлением.

Из всего вышесказанного следует, что для любого клиента должны выполняться условия $u_{c_{i,1}} \neq u_{c_{i,2}}$, так как иначе во время одновременного обновления обоих дата-центров, компания будет не в состоянии обеспечить клиенту доступ к его данным.

В связи с переводом часов в разных странах и городах мира, время обновления в некоторых дата-центрах может сдвинуться на один час вперёд. Для подготовки к непредвиденным ситуациям руководство компании хочет провести учения, в ходе которых будет выбрано некоторое непустое подмножество дата-центров, и время обновления каждого из них будет сдвинуто на один час позже внутри суток (то есть, если $u_j = h - 1$, то новым часом обновления будет 0, иначе новым часом обновления станет $u_j + 1$). При этом учения не должны нарушать гарантии доступности, то есть, после смены графика обновления должно по-прежнему выполняться условие, что данные любого клиента доступны хотя бы в одном экземпляре в любой час.

Учения — полезное мероприятие, но трудоёмкое и затратное, поэтому руководство компании обратилось к вам за помощью в определении минимального по размеру непустого подходящего подмножества дата-центров, чтобы провести учения только на этом подмножестве.

Формат входных данных

В первой строке находятся три целых числа n, m и h ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 100\,000$, $2 \leqslant h \leqslant 100\,000$) — число дата-центров компании, число клиентов компании и количество часов в сутках.

Во второй строке вам даны n чисел $u_1, u_2, \ldots, u_n$ ($0 \leqslant u_j < h$), j-е из которых задаёт номер часа, в который происходит плановое обновление программного обеспечения на компьютерах дата-центра j.

Далее в m строках находятся пары чисел $c_{i,1}$ и $c_{i,2}$ ($1 \leqslant c_{i,1}, c_{i,2} \leqslant n$, $c_{i,1} \neq c_{i,2}$), задающие номера дата-центров, на которых находятся данные клиента i.

Гарантируется, что при заданном расписании обновлений в дата-центрах любому клиенту в любой момент доступна хотя бы одна копия его данных.

Формат выходных данных

В первой строке выведите минимальное количество дата-центров k ($1 \leqslant k \leqslant n$), которые должны затронуть учения, чтобы не потерять гарантию доступности. Во второй строке выведите k различных целых чисел — номера кластеров $x_1, x_2, \ldots, x_k$ ($1 \leqslant x_i \leqslant n$), на которых в рамках учений обновления станут проводиться на час позже. Номера кластеров можно выводить в любом порядке.

Если возможных ответов несколько, разрешается вывести любой из них. Гарантируется, что хотя бы один ответ, удовлетворяющий условиям задачи, существует.

Примеры

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

Пояснение

Рассмотрим первый тест из условия. Приведённый ответ является единственным способом провести учения, затронув только один дата-центр. В таком сценарии третий сервер начинает обновляться в первый час дня, и никакие два сервера, хранящие данные одного и того же пользователя, не обновляются в один и тот же час.

С другой стороны, например, сдвинуть только время обновления первого сервера на один час вперёд нельзя — в таком случае данные пользователей 1 и 3 будут недоступны в течение нулевого часа.

Система оценки

Тесты к этой задаче состоят из трёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов предыдущих групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Дополнительные ограничения		Комментарий
		n	m	
0	0	–	–	Тесты из условия
1	30	$n \leqslant 10$	$m \leqslant 100$	–
2	30	$n \leqslant 500$	$m \leqslant 1000$	–
3	40	–	–	**Offline-проверка**

Задача Michelangelo. Отбой

Имя входного файла:	`input.txt` или стандартный поток ввода
Имя выходного файла:	`output.txt` или стандартный поток вывода
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Преподаватели Летней Какой-нибудь Школы укладывают учащихся спать.

Домик состоит из n комнат, в каждой из которых должны жить ровно b школьников. Однако к моменту отбоя оказалось, что далеко не каждый школьник находится в своей комнате. Комнаты расположены в ряд и пронумерованы по порядку от 1 до n, изначально в i-й комнате находятся a_i школьников. При этом, в домике находятся все проживающие в нём школьники и только они, то есть, $a_1 + a_2 + \ldots + a_n = nb$. В домике также живут p преподавателей, причём $p \leqslant 2$.

Процесс укладывания школьников спать устроен следующим образом. Один преподаватель встаёт у комнаты 1 и движется в направлении комнаты n. Если есть второй преподаватель, то он встаёт у комнаты n и движется в сторону комнаты 1. Закончив укладывать очередную комнату, преподаватель переходит к следующей, причём если преподавателей двое, то они входят и выходят из своих комнат одновременно. Если n нечётно, а преподавателей двое, то среднюю комнату укладывает спать только первый преподаватель. Как только все школьники уложены спать, процесс заканчивается.

Когда преподаватель заходит в комнату, он считает количество школьников внутри, затем тушит свет и закрывает комнату. Дополнительно, если количество школьников внутри не совпадает с b, этот преподаватель записывает номер комнаты в свою записную книжку и всё равно тушит свет и закрывает комнату. Преподаватели очень торопятся составить учебный план на завтра, поэтому игнорируют, кто именно находится в комнате, считая только количество школьников.

Пока преподаватели находятся в комнатах, школьники могут перебегать между ещё не закрытыми и не укладываемыми прямо сейчас комнатами, причём школьник успевает перебежать не более чем на d комнат (то есть, перемещается в комнату с номером, который отличается не более чем на d). Также, после (или вместо) перемещения любой школьник может спрятаться под кровать в той комнате, в которой находится, тогда преподаватель его не заметит при подсчёте. В любой комнате под кроватями могут спрятаться сколько угодно школьников одновременно.

Формально говоря, происходит следующий процесс:

- Объявляется отбой, к этому моменту в комнате i находится a_i школьников.
- Каждый школьник может переместиться в другую комнату, но не далее, чем на d от изначального положения, либо остаться в текущей комнате. После этого желающие спрятаться под кровать прячутся.
- В комнату 1 заходит преподаватель (а также в комнату n, если преподавателей двое), укладывает спать всех находящимся там школьников и запирает комнату (после этого нельзя ни войти в комнату, ни выйти из неё).
- Школьники из комнат с номерами от 2 до n (или до $n-1$, если преподавателей двое) могут переместиться в другие комнаты, убегая не далее, чем на d комнат от своего **текущего** местоположения, либо остаться в текущей комнате. Желающие спрятаться под кровать прячутся.
- Из комнаты 1 в комнату 2 (и, возможно, из комнаты n в комнату $n-1$) переходит преподаватель.
- Процесс продолжается, пока во всех комнатах школьники не будут уложены спать.

Обозначим за x_i количество комнат с видимым нарушением численности, которые были записаны i-м преподавателем. Школьники знают, что директор смены будет слушать жалобы на безобразие во время отбоя не более, чем от одного преподавателя домика, поэтому хотят действовать таким образом, чтобы максимальное из чисел x_i было как можно меньше. Помогите им определить, какое минимальное значение эта величина может принимать при оптимальных действиях школьников.

Формат входных данных

В первой строке записаны четыре целых числа p, n, d и b ($1 \leqslant p \leqslant 2$, $2 \leqslant n \leqslant 100\,000$, $1 \leqslant d \leqslant n-1$, $1 \leqslant b \leqslant 10\,000$) — количество преподавателей, количество комнат в домике, максимальное количество комнат, которые успевает пробежать школьник, пока преподавателя нет в коридоре, и требуемое количество школьников в одной комнате.

Во второй находятся n целых чисел $a_1, a_2, \ldots, a_n$ ($0 \leqslant a_i \leqslant 10^9$), i-е из которых соответствует количеству школьников в i-й комнате перед объявлением отбоя.

Гарантируется, что $a_1 + a_2 + \ldots + a_n = nb$.

Формат выходных данных

Выведите одно число — минимальное возможное значение максимального x_i.

Примеры

ввод	вывод
1 5 3 1 0 0 0 5 0	0
1 5 3 10 5 1 1 1 42	1
2 5 1 1 1 0 0 0 4	1
2 6 1 2 3 8 0 1 0 0	2

Пояснение

В первом примере школьники достаточно быстро бегают, чтобы разбежаться по своим комнатам ещё до того, как преподаватель войдёт в первую, тем самым ответ 0.

Во втором примере преподаватель запишет минимум одну комнату в свою записную книжку. Одной из оптимальных стратегий является следующая: перед тем, как преподаватель войдёт в первую комнату, из пятой комнаты по 10 школьников должны перебежать в комнаты 2, 3 и 4. Далее в комнатах 2, 3, 4 по одному школьнику прячется под кровать, в комнате 5 под кровать прячутся два школьника, после чего школьники не производят никаких действий. При такой последовательности действий школьников преподаватель занесёт в записную книжку только комнату 1.

В третьем примере первые три комнаты посетит первый преподаватель, а последние две — второй. Одним из оптимальных решений является следующее: на первом шаге три школьника должны перебежать из комнаты 5 в комнату 4, на втором шаге два из них должны перебежать в комнату 3 и одному из них следует там спрятаться. Тем самым недовольство первого преподавателя вызовет только комната 2, а второй преподаватель и вовсе не встретит никаких нарушений.

В четвёртом примере одной из оптимальных стратегий является следующая: на первом шаге все школьники из первой комнаты прячутся, а все школьники из второй комнаты перебегают в третью. На втором шаге один школьник перебегает из третьей комнаты в четвёртую, а ещё 5 прячутся. Тем самым первый преподаватель будет недоволен комнатами 1 и 2, а второй — комнатами 5 и 6.

Система оценки

Тесты к этой задаче состоят из семи групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, прохождение тестов из условия **не требуется** для принятия на проверку. **Offline**-проверка означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Дополнительные ограничения			Необх. группы	Комментарий
		p	n	b		
0	0	–	–	–	–	Тесты из условия
1	10	$p = 1$	$n \leqslant 1000$	$b = 1$	–	–
2	10	$p = 1$	$n \leqslant 1000$	—	1	–
3	10	—	$n \leqslant 100$	$b = 1$	–	–
4	15	—	$n \leqslant 100$	$b \leqslant 30$	0, 3	–
5	20	—	$n \leqslant 1000$	$b \leqslant 30$	0, 1, 3, 4	–
6	10	$p = 1$	—	—	1, 2	**Offline-проверка**
7	25	—	—	—	0–6	**Offline-проверка**

Задача Munch. Двоичные карты

Имя входного файла:	`input.txt` или стандартный поток ввода
Имя выходного файла:	`output.txt` или стандартный поток вывода
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Никогда не поздно научиться играть в увлекательную игру под названием «Двоичные карты»!

В игре участвует неограниченное число игральных карт разных положительных и отрицательных достоинств. Абсолютная величина, написанная на любой карте, является степенью двойки, то есть на карте может быть написано либо 2^k, либо -2^k для некоторого целого $k \geqslant 0$. Карты любого достоинства имеются в неограниченном количестве.

В начале игрок выбирает себе колоду — некоторое множество (возможно пустое) игральных карт. При этом в колоду разрешается включить произвольное количество карт каждого из достоинств, но уровень игрока оценивают по тому, насколько мало карт он включает в свою колоду. Игра состоит из n конов. На i-м кону жюри называет игроку целое число a_i. После этого игрок обязан выложить на стол такое подмножество карт из своей колоды, что сумма достоинств этих карт в точности равна a_i (можно не выкладывать никакие карты вообще, тогда сумма считается равной нулю). Если игрок не может выложить нужный набор карт, он считается проигравшим, а игра заканчивается. В противном случае игрок возвращает выложенные карты назад в свою колоду и переходит к следующему кону. Игрок считается победителем, если он на каждом кону смог выложить необходимые карты.

До вас дошли слухи, какие числа a_i жюри собирается назвать на каждом кону. Теперь вы хотите выбрать колоду с минимальным количеством карт, которая позволит вам выиграть в «Двоичные карты».

Формат входных данных

В первой строке находится целое число n ($1 \leqslant n \leqslant 100\,000$) — количество названных чисел в игре.

Во второй строке находятся n целых чисел $a_1, a_2, \ldots, a_n$ ($-100\,000 \leqslant a_i \leqslant 100\,000$) — числа, называемые на каждом кону.

Формат выходных данных

В первой строке выведите число k ($0 \leqslant k \leqslant 100\,000$) — минимальное количество карт, которые нужно выбрать в колоду, чтобы выиграть в «Двоичные карты».

Во второй строке выведите k целых чисел $b_1, b_2, \ldots, b_k$ ($-2^{20} \leqslant b_i \leqslant 2^{20}$, $|b_i|$ является степенью двойки) — достоинства карт, составляющих колоду. Достоинства можно выводить в любом порядке. Если существует несколько колод оптимального размера, разрешается вывести любую из них.

Гарантируется, что существует колода карт минимального размера, удовлетворяющая требованиям на величины чисел выше.

Примеры

ввод	вывод
1 9	2 1 8
5 -1 3 0 4 7	3 4 -1 4
4 2 -2 14 18	3 -2 2 16

Пояснение

В первом тесте из условия на единственном кону можно положить обе карты из колоды. Обратите внимание, этот тест из условия — единственный из подходящих под ограничения первой группы тестов.

Во втором тесте из условия в первый кон можно выложить единственную карту -1, во второй кон — карты 4 и -1, в третий кон можно не выкладывать ничего, в четвёртый кон — только карту 4, а в пятый кон — всю колоду.

Система оценки

Тесты к этой задаче состоят из четырнадцати групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов всех предыдущих групп. Обратите внимание, прохождение тестов из условия **не требуется** для принятия на проверку. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Дополнительные ограничения		Комментарий
		n	$\lvert a_i\rvert$	
0	0	–	–	Тесты из условия
1	8	$n = 1$	$\lvert a_i\rvert \leqslant 10$	–
2	7	$n \leqslant 10$	$\lvert a_i\rvert \leqslant 10$	–
3	7	$n \leqslant 30$	$\lvert a_i\rvert \leqslant 30$	–
4	7	$n \leqslant 50$	$\lvert a_i\rvert \leqslant 50$	–
5	7	$n \leqslant 100$	$\lvert a_i\rvert \leqslant 100$	–
6	7	$n \leqslant 300$	$\lvert a_i\rvert \leqslant 300$	–
7	8	$n \leqslant 500$	$\lvert a_i\rvert \leqslant 500$	–
8	7	$n \leqslant 1000$	$\lvert a_i\rvert \leqslant 1000$	**Offline-проверка**
9	6	$n \leqslant 3000$	$\lvert a_i\rvert \leqslant 3000$	**Offline-проверка**
10	7	$n \leqslant 5000$	$\lvert a_i\rvert \leqslant 5000$	**Offline-проверка**
11	6	$n \leqslant 10\,000$	$\lvert a_i\rvert \leqslant 10\,000$	**Offline-проверка**
12	7	$n \leqslant 30\,000$	$\lvert a_i\rvert \leqslant 30\,000$	**Offline-проверка**
13	7	$n \leqslant 50\,000$	$\lvert a_i\rvert \leqslant 50\,000$	**Offline-проверка**
14	9	–	–	**Offline-проверка**