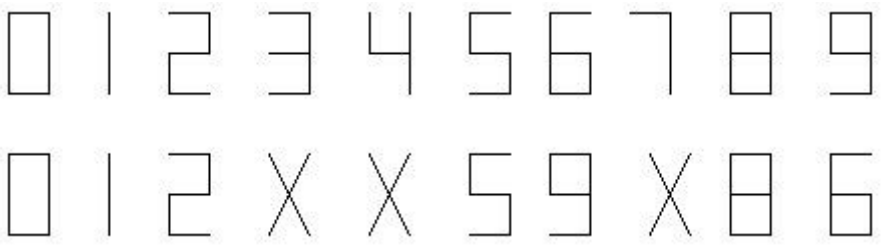
A. Chess Clock

Ограничения: время работы — 1 секунда, использование памяти — 256МБ

Команда экспертов ACM, работающая над созданием Больших Шахмат, столкнулась с необходимостью разработки специальных часов для проведения турниров. Доска для игры в Большие Шахматы может оказаться такой большой, что турнирные часы не поместятся на столе, и судья будет держать их в руках. Тогда может возникнуть ситуация случайного переворота часов («верх» и «низ» часов неразличимы). На рисунке ниже показано, что получается при перевороте каждой цифры на электронном табло. Символ «X» означает, что корректной цифры не получается.



Назовем момент времени «интересным», если при перевороте часов в этот момент на табло также отображается корректное время суток. К примеру, моменты 22:11 и 21:51 являются «интересными» (при перевороте на табло отображаются 11:22 и 15:12 соответственно), а момент 22:19 не является «интересным» (при перевороте получается 61:22).

К тому же необходимо учесть, что игра в Большие Шахматы может затянуться (ее максимальная продолжительность, согласно правилам, составляет 23:59, иначе соперникам присуждается ничья) и окончится в следующие сутки, то есть, после 00:00.

Помогите экспертам ACM определить количество «интересных» моментов в заданном интервале времени.

Входные данные

В первой строке ввода задается число тестов T ($1 \le T \le 99$). Каждый тест описывается строкой «HH:MM – HH:MM» (без кавычек), содержащей начало и конец интервала времени, где HH — целое число часов от 00 до 23 включительно, а MM — целое число минут от 00 до 59 включительно.

Выходные данные

Для каждого теста в отдельной строке вывести количество «интересных» моментов времени.

Пример

Входные данные	Результат
1 21:00 - 21:13	6

B. Сжатие диплома

Ограничения: время работы — 1 секунда, использование памяти — 256МБ

Феде необходимо до завтра сдать диплом на проверку. Сама записка уже полностью написана, но Федя столкнулся с проблемой — по требованиям дипломного руководителя длина текста записки не должна превышать K символов. Ему необходимо в срочном порядке сократить текст диплома так, чтобы он соответствовал требованию по длине.

К счастью, в пояснительной записке можно использовать добавочные материалы: таблицы, рисунки, список сокращений. Размер текста в добавочных материалах не учитывается при подсчете длины диплома. Отсутствием ограничений на добавочный материал Федя принялся решать эту проблему. А именно — использовать сокращения, чтобы уменьшить объем текста диплома.

Текст диплома представляет собой строку, содержащую только маленькие латинские литеры. Сокращение представляет собой слово (аббревиатуру), которое заменяет собой другое слово в тексте. Аббревиатура в тексте должна быть заключена в квадратные скобки для возможности обратной расшифровки. Квадратные скобки учитываются при подсчете длины. **Словом в тексте служит любая подстрока.**

Например, текст «ababxabab» может быть заменен на «[s]x[s]», где «s → abab». Таким образом, мы смогли уменьшить длину текста с 9 до 7.

Помогите Феде найти такой список сокращений, при котором пояснительная записка к диплому будет принята дипломным руководителем.

Входные данные

В первой строке теста содержится строка S ($1 \le len(S) \le 10^5$) — текст пояснительной записки к диплому. В следующей строке содержится число K ($1 \le K \le 10^5$) — требование к максимальной длине пояснительной записки.

Выходные данные

Если невозможно найти такой список сокращений, который бы сократил записку до нужной длины, выведите «NO».

Иначе выведите «YES», затем с новой строки сокращенный текст пояснительной записки. В следующей строке выведите число Q — количество сокращений, которые пойдут в добавочный материал. В последующих Q строках выведите аббревиатуру и исходное слово через пробел.

Пример

Входные данные	Результат
diploma 10	YES diploma 0
abbreviationmusthaveanabbreviation 20	YES [x]musthavean[x] 1 x abbreviation

C. Lorem ipsum

Ограничение на время работы: для C и C++ — 1 секунда, для Java — 2 секунды
Ограничение на использование памяти: 256МБ

Вася Пупкин — тот самый студент X-OSPU (eXtreme Open Source Programming University), задачи про которого вы решаете уже пятый год — в этом году уже пятикурсник. А это значит, что ему пора писать диплом. Ну как, «пора»… Пора было пару месяцев назад, а завтра у него уже малая защита. На первый взгляд у Васи всё готово, но есть одно «но». В процессе работы над пояснительной запиской ему было лень писать некоторые части текста, и он просто оставлял на страницах пустые места. «Потом напишу», — думал Вася. Ну, вот «потом» и наступило.

Вася просмотрел все N страниц диплома и подсчитал, что на i-ой странице не дописано w_i слов. «За одну ночь всё дописать не успею», — трезво оценил ситуацию Вася. Находясь в полном отчаянье и почему-то не подумав посоветоваться со своим научным руководителем, Вася придумал, как ему в тот момент показалось, не самый плохой выход. «А заполню-ка я пустые места текстом lorem ipsum. Авось, — наивно подумал Вася, — на малой защите никто так внимательно вчитываться не будет. Ну а к большой защите я обязательно всё допишу», — пообещал он себе в $(w_1+w_2+...+w_N+1)$-ый раз.

Конечно, Вася понимал, что если его маленькая хитрость всё же будет обнаружена, то ему, мягко говоря, несдобровать. Поэтому для минимизации вероятности столь неблагоприятного исхода он решил пользоваться следующим правилом: «Я не буду заполнять lorem ipsum пустые места на всех страницах. Выберу часть страниц и заполню только на них. Но на выбранных страницах заполню все пустые места — тут уж терять нечего. А пустые места на остальных страницах тогда нужно дописать сегодня. И я буду делать отступы между выбранными страницами: если я выбрал страницу с чётным номером, то я не буду выбирать M_e страниц до и после неё, а если с нечётным номером — то не буду выбирать M_o страниц до и после». Поскольку, как известно каждому студенту, последняя ночь перед сдачей чего-нибудь — самая короткая, выбрать страницы нужно так, чтобы дописывать сегодня пришлось как можно меньше.

Помогите Васе оценить, сколько слов он может заполнить текстом lorem ipsum.

Входные данные

В первой строке находится три числа, разделённых пробелами: N ($1 \le N \le 10^4$) — количество страниц в дипломе, M_e и M_o ($0 \le M_e, M_o \le 10^4$) — сколько страниц нельзя выбирать для заполнения текстом lorem ipsum вокруг выбранной чётной и нечётной страницы соответственно.

Во второй строке находится N чисел w_i ($0 \le w_i \le 10^9$), разделённых пробелами, — количество недописанных слов на каждой из страниц. Страницы нумеруются от 1.

Выходные данные

Выведите одно число — максимальное количество недописанных слов, которое Вася может заполнить текстом lorem ipsum, придерживаясь своего правила.

Пример

Входные данные	Результат
7 1 2 7 2 3 9 5 6 1	22

В этом примере следует выбрать страницы 1, 4 и 6, что позволит заполнить lorem ipsum 22 слова.

D. Примечательные квадраты

Ограничения: время работы — 1 секунда, использование памяти — 256МБ

Найти n-разрядные целые числа (n — четное), которые являются квадратами некоторых целых чисел, и у которых первая цифра равна второй, третья равна четвертой и так далее.

Входные данные

Одно число n ($2 \le n \le 18$) — разрядность числа. Число n — четное.

Выходные данные

Вывести найденные числа и квадратные корни из них через пробел, каждое — в отдельной строке, по возрастанию. Если числа не найдены, вывести -1.

Пример

Входные данные	Результат
4	7744 88

E. Мудреная заливка

Ограничение на время работы: для C и C++ — 3 секунды, для Java — 15 секунд
Ограничение на использование памяти: 256МБ

Все знают Леху из АС-182. А кто не знает, тот скоро определенно узнает, ведь у Лехи есть мечта — получить мировое признание в качестве гениального художника. Только вот рисовать Леха не умеет, зато вполне сносно пишет плагины для графического редактора Above Photoshow GG.

На носу ежегодная выставка студенческих картин в ОНПУ. К ней Леха готовит нечто особенное — целую серию картин размера $m \times n$ в стиле трех-битного криптоабстракционизма. Все его картины представлены набором из трех цветов и их сочетаний, описанных следующей таблицей.

Кодировка	R	Y	B	O = R + Y	V = R + B	G = B + Y	M = R + Y + B	_ (подчёркивание)
Цвет	Красный	Желтый	Синий	Оранжевый	Фиолетовый	Зелёный	Грязный	Нет цвета

К сожалению, картины далеки от совершенства и требуют значительной постобработки чтобы стать настоящим произведением искусства. Достичь совершенства можно лишь с помощью мудреной заливки, применив к каждой картине набор из k операций добавления «+» и удаления «-» цвета по заданным координатам $(x; y)$.

Операция добавления смешивает цвет пикселя по указанным координатам с заданным цветом и распространяется в соседние пиксели по горизонтали и вертикали, если верно любое из условий:

1. Цвет пикселя совпадает с начальным цветом пикселя, с которого началась заливка.
2. У пикселя отсутствует цвет.
3. Цвет пикселя совпадает с цветом операции заливки.

Операция удаления цвета удаляет из ячейки заданный цвет и распространяется в соседние пиксели по горизонтали и вертикали, если выполняется любое из условий (стартовая ячейка тоже подлежит проверке):

1. Пиксель содержит в себе заданный цвет.
2. Был задан цвет «M», и цвет пикселя совпадает с начальным цветом пикселя, с которого началась заливка.

Помогите Лёхе реализовать реализовать его мудрёную заливку.

Входные данные

В первой строке дано два числа m и n ($1 \le m, n \le 1000$) — размер картины по горизонтали и вертикали. За ними следует n строк по m символов, где каждый символ представляет собой закодированный цвет.

Следующей строкой дано число k ($1 \le k \le 10$) — количество операций над картиной, а за ним следует k строк формата «x y o c», где x и y представляют собой координаты точки на рисунке ($0 \le x \le m-1$, $0 \le y \le n-1$), o принимает значение «+» для операции добавление и «-» для операции удаления, а c является символом в котором закодирован цвет операции.

Выходные данные

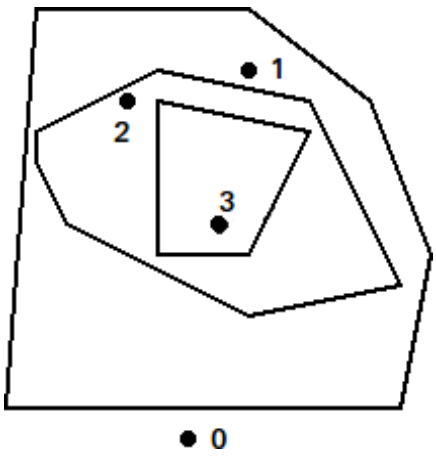
Выведите n строк по m символов описывающих картину после произведения всех операций над ней в порядке установленном вводом.

Пример

Входные данные	Результат
10 8 YYYYYBBBBB YYYYYBBBBB YYYYYBBBBB YYYYYBBBBB OOOOOGGGGG OOOOOGGGGG OOOOOGGGGG OOOOOGGGGG 5 0 7 - Y 0 0 + B 5 5 + R 3 6 - R 0 7 + Y	BBBBBBBBBB BBBBBBBBBB BBBBBBBBBB BBBBBBBBBB YYYYYBBBBB YYYYYBBBBB YYYYYBBBBB YYYYYBBBBB

F. Тир

Ограничение на время работы: для C и C++ — 3 секунды, для Java — 10 секунд
Ограничение на использование памяти: 256МБ



На следующих выходных в местном парке открывается новый тир. Его особенностью будут необычные формы мишеней, представляющие собой выпуклые многоугольники.

Мишень состоит из N многоугольников, причем только один из них является внешним, а все остальные строго внутри какого-то другого. Каждый многоугольник может либо содержать в себе какой-то другой, и только один, либо не содержать ни одного. Это означает, что многоугольники представляют собой некую цепочку вложенности.

Руководство тира решило оптимизировать свою работу, а именно — автоматизировать подсчет очков после каждого выстрела. Для этого они нанимают вас, как опытного программиста.

Количество очков после каждого выстрела равняется количеству многоугольников, которые были задеты выстрелом. Если точка $P(x, y)$ является позицией прохождения пули сквозь мишень, то считаются задетыми только те многоугольники, у которых точка P находится строго внутри.

Входные данные

В первой строке находится число N ($1 \le N \le 10^5$) — количество многоугольников, из которых состоит мишень. Далее находятся описания N многоугольников. Описание многоугольника состоит из числа M_i ($3 \le M_i \le 20$) — количество вершин, а также описания M_i вершин, каждая из которых описана двумя числами x и y ($-10^9 \le x, y \le 10^9$).

Далее идет число Q ($1 \le Q \le 10^5$) — количество выстрелов. В следующих Q строках содержится по одной точке, которая описана двумя числами x и y ($-10^9 \le x, y \le 10^9$) — координаты прохождения пули.

Выходные данные

Выведите Q чисел – количество очков за каждый выстрел.

Пример

Входные данные	Результат
2 4 -1 -1 1 -1 1 1 -1 1 4 -2 -2 2 -3 3 2 -2 3 2 2 -1 -3 -1	1 0

G. Еремей спешит домой

Ограничение на время работы: для C и C++ — 1 секунда, для Java — 15 секунд
Ограничение на использование памяти: 256МБ

Однажды Еремей со своими друзьями пошел гулять по лабиринту, который состоит из комнат и дверей между ними. Из любой комнаты есть до четырех дверей, ведущих в соседние комнаты. Чтобы не заблудиться, никто не откалывался от коллектива дальше, чем на соседнюю комнату. Иными словами, любой человек может дойти до любого другого, не посещая пустые комнаты. Но уже слишком поздно и всем пора идти домой. Еремей дал команду всем своим друзьям собраться в той комнате, где он находится. Надо сделать это как можно быстрее, ведь уже начинает темнеть. Одновременно в лабиринте не может быть открыто больше одной двери, поэтому надо собраться в комнате, где есть Еремей, сделав при этом как можно меньше открытий дверей. Помогите Еремею и компании, иначе их родители будут сильно переживать.

Входные данные

В первой строке дано два числа N и M ($1 \le N, M \le 300$) — размеры лабиринта.

В следующих N строках записано по M чисел от -1 до 100 — количество людей в комнате, или -1 если в эту комнату заходить нельзя.

Далее следует два числа R и C ($1 \le R \le N$, $1 \le C \le M$), задающих комнату (строку и столбец соответственно), в которой находится Еремей. Гарантируется, что в ней есть хотя бы 1 человек.

Выходные данные

В первой строке выведите число K — минимальное количество команд, которые нужно выполнить, чтоб все собрались в комнате с Еремеем.

В следующих K строках выведите K команд вида «R1 C1 R2 C2 P», что означает, что нужно открыть дверь между комнатами $(R1, C1)$ и $(R2, C2)$, переместить из первой во вторую P человек и закрыть дверь.

Если ответов несколько — выведите любой.

Пример

Входные данные	Результат
4 5 -1 1 0 0 3 -1 4 3 8 2 2 4 -1 3 9 1 3 5 9 -1 2 2	13 1 2 2 2 1 1 5 2 5 3 2 5 2 4 5 2 4 2 3 13 2 3 2 2 16 3 5 3 4 9 3 4 4 4 12 4 4 4 3 21 4 3 4 2 26 4 2 3 2 29 4 1 3 1 1 3 1 3 2 3 3 2 2 2 36

H. Диплом

Ограничения: время работы — 1 секунда, использование памяти — 256МБ

Помните Васю Пупкина? Ещё недавно он только поступил в X-OSPU (eXtreme Open Source Programming University), а вот прямо сейчас он уже несёт свой диплом на переплёт. Но совсем не о дипломе думает Вася. Только что на его смартфон пришло уведомление о том, что доступно для загрузки долгожданное обновление до RobotOS 9.0 Pumpkin. «Не буду ждать возвращения домой, поставлю прямо сейчас», — решил Вася.

И вот, Вася идёт, неся в одной руке пачку только что распечатанных страниц диплома, а другой рукой запускает процесс обновления. И в этот момент Вася в 14 835-ый раз в своей жизни вспомнил, почему он уже 14 834 раза обещал себе больше никогда не залипать в телефон на ходу. Вася бежал наперегонки с ветром, ловил страницы руками, придавливал ногами, сбивал с ног идущих по дороге студентов, едва разминулся с деканом…

Когда погоня наконец была закончена, и Васе удалось собрать все страницы, перед ним оказалась стопка из N страниц диплома, пронумерованных в строгом порядке от 1 до N, но разложенных в строгом беспорядке. Помогите Васе отсортировать страницы в правильном порядке, чтобы он мог поскорее начать установку долгожданного обновления для своего смартфона.

Входные данные

В первой строке дано число N ($1 \le N \le 10^3$) — количество страниц в дипломе Васи. Во второй строке дано N чисел, разделённых пробелами — номера страниц, в том порядке, в котором они оказались после окончания погони.

Выходные данные

В единственной строке выведите N чисел, разделённых пробелами — номера страниц в правильно отсортированном порядке.

Пример

Входные данные	Результат
5 3 2 4 5 1	1 2 3 4 5

I. Recipes

Ограничение на время работы: для C и C++ — 1 секунда, для Java — 3 секунды
Ограничение на использование памяти: 256МБ

Дима — хороший студент, отличник, но тем не менее, как и многие его сверстники, не отказывает себе в отдыхе и с радостью играет в Lineage2. Но безделью не победить! Дима решил извлечь что-то полезное из этого времяпровождения. Вспомнив, как лихо он научился «крафтить» в игре вещи: собирать предметы (такие как, например, камни, уголь, стекло или древесину) и делать из них вещи (такие как броня или оружие, снова же в примеру), он понял, что эти навыки могут помочь ему в реальной жизни наполнять холодильник едой. И его не остановить.

Посмотрев на книжной полке, Дима нашел книгу с рецептами, успешно перенес их в электронный вид, и теперь у него есть N рецептов. Каждый из N рецептов содержит по M_i ингредиентов, для каждого из которых имеется Q_{ij} — необходимое количество. Ингредиентом может быть как продукт из холодильника, так и приготовленное блюдо (название другого рецепта).

Но не имея знаний о том, что есть в наличии дома, электронные рецепты пригодиться не могут. Дима, как уже опытный человек, перенес в электронный вид информацию обо всем, что есть в холодильнике (а готовые блюда, как оказалось, в холодильнике тоже были). Теперь у Димы есть еще P ингредиентов, у каждого из которых R_i — их количество.

Обдумав, как он хочет реализовать свою программу, Дима проверил все имеющиеся данные. Он убедился, что рецепт на каждое блюдо существует только один, и что блюда не могут быть взаимозависимыми.

План такой — Дима в любой момент, увидев название какого-то блюда, может понять, знает ли он как его готовить и есть ли у него необходимые ингредиенты в нужном количестве. Но Дима, к сожалению, более опытен в играх, чем в программировании. Помогите Диме решить эту задачу.

Входные данные

В первой строке находится одно число: N ($1 \le N \le 100$) — количество рецептов. Далее следуют описания рецептов.

Описание каждого рецепта начинается со строки, содержащей название рецепта (строку) и количество ингредиентов M_i ($1 \le M_i \le 250$). Далее следует M_i строк, описывающих ингредиенты. Каждая строка содержит название ингредиента (строку) и требуемое количество Q_{ij} ($1 \le Q_{ij} \le 250$).

После описания рецептов в отдельной строке находится число P ($1 \le P \le 10^4$) — количество готовых ингредиентов. Затем следует P строк с описанием ингредиентов. Каждая строка содержит название ингредиента (строку) и имеющееся количество R_i ($1 \le R_i \le 5000$).

Последняя строка содержит название требуемого блюда.

Любая строка, представляющая собой название ингредиента или рецепта, содержит только английские символы в любом регистре и пробелы; имеет длину не более 1000 символов.

Гарантируется, что суммарное количество любого ингредиента, который потребуются для приготовления блюда, не превышает 2^{30}.

Выходные данные

Выведите «YES», если блюдо возможно приготовить, или «NO», если блюдо приготовить нельзя.

Пример

Входные данные	Результат
2 Tomato paste 2 Tomato 1 Onion 2 Borsch 6 Cabbage 1 Chicken drumsticks 2 Beet 1 Tomato paste 2 Carrot 1 Potatoes 2 8 Cabbage 1 Chicken drumsticks 4 Beet 3 Carrot 3 Potatoes 2 Tomato paste 1 Tomato 1 Onion 1 Borsch	NO

В данном примере для приготовления блюда «Borsch» достаточно всех ингредиентов, кроме «Tomato paste». Среди готовых «Tomato paste» содержится лишь в одном экземпляре, а для того, чтоб приготовить еще один экземпляр, нам недостаточно ингредиента «Onion».

J. Optimal Tariff

Ограничения: время работы — 1 секунда, использование памяти — 256МБ

Имеется оператор, предлагающий тарифы с заданной абонплатой. Абонплата различна для всех тарифов. Оператор на данный момент предлагает 4 тарифа:

1. абонплата 30 грн/мес., предоставляется 60 мин. внутри оператора и 30 мин. с удержанием 2 грн за каждый звонок на все остальные направления;
2. абонплата 70 грн/мес., предоставляется 120 мин. внутри оператора и 60 мин. без оплаты за звонок на все остальные направления;
3. абонплата 90 грн/мес., предоставляется 60 мин. внутри оператора и 300 мин. на все остальные направления без оплаты за звонок;
4. абонплата 250 грн/мес. — полный безлимит.

При превышении лимита минут — стоимость 2 грн/мин. с удержанием 2 грн за каждый звонок независимо внутрисетевой это звонок или нет. При исчерпании тарифных минут в течение одного звонка оплата начисляется как за звонок сверх лимита (взимается плата за сам звонок), однако дополнительно оплачиваются только минуты, потраченные сверх лимита.

Клиент предоставляет список своих звонков за предыдущий месяц (номер телефона и длительность звонка), и необходимо по представленным данным выбрать тариф, по которому клиент потратит меньше всего денег. В случае, когда нескольких тарифов одинаково выгодны, выбирать тариф с меньшим порядковым номером. Телефоны, начинающиеся с кодов 033 и 055 принадлежат оператору и разговоры с ними являются внутрисетевыми.

Входные данные

Количество записей N ($1 \le N \le 500$), затем N строк, в каждой из которых через пробел записан номер телефона, состоящий из 10 цифр и начинающийся с 0, и длительность звонка в минутах (не более 400) (например, 12).

Выходные данные

Одна строка — номер предлагаемого тарифа и рассчитанное количество денег, которые за него пришлось бы потратить, через пробел.

Пример

Входные данные	Результат
5 0345123455 12 0231474839 1 0345678900 15 0334848484 77 0552374888 34	2 70

K. Neighbor Stations

Ограничение на время работы: для C и C++ — 1 секунда, для Java — 2 секунды
Ограничение на использование памяти: 256МБ

На ограниченной территории (которую будем считать плоской) расположены N станций наблюдения за космическими объектами. Положение каждой станции как точки в прямоугольной системе координат на плоскости задано двумя целыми числами (x_i, y_i).

Назовем соседними такие две станции, расстояние между которыми не превышает заданной величины d. Между соседними станциями возможен обмен экстренными сообщениями по специальному каналу. Необходимо выбрать из данного множества станций максимально большое по числу элементов (не менее двух) подмножество такое, чтобы любая из его станций могла бы передать сообщение всем остальным его станциям (непосредственно или через другие).

Входные данные

В первой строке через пробел 2 числа: n ($2 \le n \le 1000$) — число станций, d ($1 \le d \le 4 \cdot 10^6$) — максимальное расстояние между соседними станциями. Далее следует n строк, в i-ой строке — координаты x_i и y_i i-ой станции ($-2 \cdot 10^6 \le x_i, y_i \le 2 \cdot 10^6$).

Выходные данные

Максимально возможное число элементов в выбранном подмножестве. Если такого подмножества не существует, вывести 1.

Пример

Входные данные	Результат
5 1 1 2 1 3 2 4 3 5 4 12	2
4 3 1 2 1 6 2 -2 -3 5	1
7 7 1 2 1 6 2 4 -8 2 -12 5 -11 3 -11 -3	4

В первом примере связь можно установить между станциями с координатами $(1,2)$ и $(1,3)$. Во втором примере связь установить невозможно. В третьем примере максимальное подмножество образуют 4 последние станции.

L. Губка Боб и посуда

Ограничение на время работы: для C и C++ — 1 секунда, для Java — 3 секунды
Ограничение на использование памяти: 256МБ

Однажды после празднования дня рождения Губка Боб пошел на кухню, чтобы вымыть посуду, которая осталась после поедания вкусного торта. Всего у него на празднике было N людей (включая Боба), после которых осталось N грязных тарелок.

Тарелки могут отличаться размерами. Так, тарелка под номером i имеет радиус R_i. Причем радиус может быть нулевым — вы же верите в сказки? Губка Боб очень любит порядок, в связи с чем складывает все тарелки в стопку от большей к меньшей по радиусу. К сожалению, гости сложили грязные тарелки не так, как подходит Губке Бобу.

Мытье тарелок происходит по следующему алгоритму: берется верхняя тарелка из грязной стопки, вымывается, а затем кладется на верх стопки с чистыми тарелками. Если на верху стопки чистых тарелок есть тарелки с меньшим радиусом, то все такие тарелки приподнимаются, затем ставится новая тарелка и возвращаются старые в исходном порядке. Это гарантирует, что стопка чистых тарелок всегда будет правильной. Необходимо учитывать, что за каждую приподнятую тарелку расходуется одна единица энергии.

Тарелки в исходной последовательности представлены в виде массива R, где R_i - радиус i-ой тарелки. Тарелка с номером 0 находится в самом верху стопки. Губка Боб решил перед мытьем переставить часть тарелок таким образом, чтобы минимизировать последующие затраты энергии. Он решил выбрать некоторую позицию i и переставить все тарелки выше i-ой ($0 \dots i-1$) вниз стопки без изменения их порядка так, чтобы тарелка $i-1$ стала самой нижней. Нужно найти оптимальное i. Сможете ли вы справиться с этой задачей, чтобы помочь главному герою?

Входные данные

В первой строке теста содержится число N ($1 \le N \le 10^5$) — количество тарелок. Далее идут N чисел R_i ($0 \le R_i \le 10^9$), каждое из которых описывает радиус i-ой тарелки.

Выходные данные

Два числа через пробел, где первое — это оптимальная позиция, с которой стопка должна начинаться, а второе — количество необходимой энергии. Если возможны несколько оптимальных позиций, то выберите ту, что имеет меньший индекс.

Пример

Входные данные	Результат
5 5 4 9 4 9	2 2