# Общая информация по задачам второго тура

| **Задача** | **Тип задачи** | **Ограничения** |
|---|---|---|
| 5. Расшифровка | Стандартная | 1 с, 512 МБ |
| 6. Быстрая сортировка | Стандартная | 1 с, 512 МБ |
| 7. Робомарафон | Стандартная | 1 с, 512 МБ |
| 8. Сложение без переносов | Стандартная | 2 с, 512 МБ |

Необходимо считывать данные из стандартного потока ввода. Выходные данные необходимо выводить в стандартный поток вывода.

Во всех задачах, **кроме задачи 7**, баллы за подзадачу начисляются только, если все тесты для подзадачи пройдены.

# Задача 5. Расшифровка

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Известно, что если сохранить в каждом слове текста первую и последнюю букву, а остальные переставить произвольным образом, получившийся текст по-прежнему можно достаточно свободно прочитать. В лаборатории информатики исследуют аналогичный феномен для числовых последовательностей.

Будем называть последовательность, состоящую из целых положительных чисел, *корректной*, если первое число в этой последовательности является минимальным, а последнее — максимальным. Например, последовательности $[1, 3, 2, 4]$ и $[1, 2, 1, 2]$ являются корректными, а последовательность $[1, 3, 2]$ — нет.

Задана последовательность $[a_1, a_2, \ldots, a_n]$. Будем называть отрезок элементов заданной последовательности $[a_l, a_{l+1}, \ldots, a_r]$ корректным, если он представляет собой корректную последовательность: $a_l$ является минимальным числом на этом отрезке, а $a_r$ — максимальным.

В рамках исследования необходимо разбить заданную последовательность на минимальное количество непересекающихся корректных отрезков. Например, последовательность $[2, 3, 1, 1, 5, 1]$ можно разбить на три корректных отрезка: $[2, 3]$ и $[1, 1, 5]$ и $[1]$.

Требуется написать программу, которая по заданной последовательности определяет, на какое минимальное количество корректных отрезков её можно разбить.

## Формат входных данных

Первая строка входных данных содержит целое число $n$ ($1 \leqslant n \leqslant 300\,000$) — количество элементов в заданной последовательности.

Вторая строка содержит $n$ целых чисел $a_1, a_2, \ldots, a_n$ — заданную последовательность ($1 \leqslant a_i \leqslant 10^9$).

## Формат выходных данных

Выведите одно число — минимальное количество корректных отрезков, на которое можно разбить заданную последовательность.

## Примеры

| стандартный ввод | стандартный вывод |
|---|---|
| 5<br>5 4 3 2 1 | 5 |
| 4<br>1 3 2 4 | 1 |
| 6<br>2 3 1 1 5 1 | 3 |

## Система оценивания

| Подзадача | Баллы | Ограничения | Необходимые подзадачи | Результаты во время тура |
|---|---|---|---|---|
| | | $n$ | | |
| 1 | 30 | $n \leqslant 500$ | У | Потестовые |
| 2 | 30 | $n \leqslant 5000$ | У, 1 | Потестовые |
| 3 | 40 | $n \leqslant 300\,000$ | У, 1, 2 | Первая ошибка |

# Задача 6. Быстрая сортировка

| | |
|---|---|
| Ограничение по времени: | 1 секунда |
| Ограничение по памяти: | 512 мегабайт |

Новый процессор UL-2018, разработанный в научной лаборатории, предназначен для быстрой обработки массивов. Ключевой особенностью архитектуры нового процессора является операция *расслоения* отрезка массива.

Рассмотрим массив $[a_1, a_2, \ldots, a_n]$. Операция расслоения характеризуется двумя целыми числами $l$ и $r$ — номером первого и последнего элемента отрезка массива, к которому она применяется. Обозначим операцию расслоения отрезка массива $[a_l, a_{l+1}, \ldots, a_r]$ как $S(l, r)$. После выполнения операции $S(l, r)$ элементы массива на этом отрезке переупорядочиваются следующим образом. Сначала располагаются элементы отрезка с позиций $a_{l+1}, a_{l+3}, \ldots$, то есть элементы с позиций $i$, для которых значение $i - l$ нечетно, их относительный порядок остаётся неизменным. Затем идут элементы отрезка $a_l, a_{l+2}, \ldots$, то есть элементы с позиций $i$, для которых значение $i - l$ четно, они также сохраняют свой относительный порядок.

Например, рассмотрим массив $[2, 4, 1, 5, 3, 6, 7, 8]$. После выполнения операции расслоения $S(2, 6)$ изменится порядок элементов на отрезке массива $[4, 1, 5, 3, 6]$. Новый порядок элементов на этом отрезке следующий: $[1, 3, 4, 5, 6]$, а весь массив после выполнения этой операции равен $[2, 1, 3, 4, 5, 6, 7, 8]$.

Для демонстрации возможностей нового процессора решено было использовать операцию расслоения для реализации алгоритма сортировки массива различных чисел. Задан массив, содержащий $n$ элементов, $1 \leqslant n \leqslant 3000$. Элементы массива — различные целые числа от 1 до $n$. Необходимо отсортировать заданный массив по возрастанию, используя не более 15 000 операций расслоения отрезка массива.

Например, приведенный выше массив $[2, 4, 1, 5, 3, 6, 7, 8]$ можно отсортировать, используя две операции расслоения. Сначала выполним продемонстрированную выше операцию $S(2, 6)$, после которой массив принимает вид $[2, 1, 3, 4, 5, 6, 7, 8]$, а затем операцию $S(1, 2)$, массив примет вид $[1, 2, 3, 4, 5, 6, 7, 8]$.

Требуется написать программу, которая по заданному массиву определит последовательность не более, чем из 15 000 операций расслоения, после выполнения которых заданный массив окажется отсортирован по возрастанию. Минимизировать количество выполненных операций расслоения не требуется, достаточно использовать не более 15 000 операций.

## Формат входных данных

Первая строка входных данных содержит целое число $n$ — количество элементов в заданном массиве ($1 \leqslant n \leqslant 3000$).

Вторая строка содержит $n$ целых чисел $a_1, a_2, \ldots, a_n$ — элементы заданного массива ($1 \leqslant a_i \leqslant n$, все $a_i$ различны).

## Формат выходных данных

В первой строке выходных данных необходимо вывести целое число $k$ — количество выполненных операций расслоения ($0 \leqslant k \leqslant 15\,000$).

В последующих $k$ строках необходимо вывести описание самих операций, в порядке их выполнения. Каждая операция описывается двумя целыми числами $l$ и $r$ — номером первого и последнего элемента отрезка массива, к которому необходимо применить очередную операцию расслоения ($1 \leqslant l < r \leqslant n$).

Следует обратить внимание, что минимизировать число $k$ не требуется. Из возможных последовательностей операций расслоения, содержащих не более 15 000 операций, после выполнения которых заданный массив будет отсортирован по возрастанию, можно вывести любую. Гарантируется, что хотя бы одна такая последовательность существует.

## Примеры

| стандартный ввод | стандартный вывод |
|---|---|
| 5<br>3 1 4 2 5 | 1<br>1 5 |
| 8<br>2 4 1 5 3 6 7 8 | 2<br>2 6<br>1 2 |
| 2<br>2 1 | 3<br>1 1<br>2 2<br>1 2 |

## Пояснение к примеру

Второй тест из примера подробно разобран в условии задачи.

Для третьего теста из примера существует решение из одной операции расслоения, но поскольку минимизировать количество операций не требуется, приведенный в примере ответ также является правильным.

## Система оценивания

| Подзадача | Баллы | Ограничения | | Необходимые подзадачи | Результаты во время тура |
|---|---|---|---|---|---|
| | | $n$ | Доп. ограничения | | |
| 1 | 20 | $1 \leqslant n \leqslant 100$ | Существует ответ с $k = 1$ | | Потестовые |
| 2 | 30 | $1 \leqslant n \leqslant 100$ | | У, 1 | Потестовые |
| 3 | 30 | $1 \leqslant n \leqslant 1000$ | | У, 1, 2 | Первая ошибка |
| 4 | 20 | $1 \leqslant n \leqslant 3000$ | | У, 1 – 3 | Первая ошибка |

# Задача 7. Робомарафон

| | |
|---|---|
| Ограничение по времени: | 3 секунды |
| Ограничение по памяти: | 512 мегабайт |

В робомарафоне принимают участие $n$ роботов. Роботы должны преодолеть одинаковую дистанцию, передвигаясь по расположенным рядом друг с другом дорожкам шириной один метр каждая. Известно, что расположенный на $i$-й дорожке робот преодолевает дистанцию за $a_i$ секунд.

В точке старта каждого робота установлено специальное сигнальное устройство, которое должно сработать в момент старта. Чтобы сделать соревнования менее предсказуемыми, судьи перед стартом могут отключить некоторые сигнальные устройства, остальные устройства останутся *активными*. Только активные устройства срабатывают в тот момент, когда главный судья начинает робомарафон. В начале робомарафона хотя бы одно сигнальное устройство должно являться активным.

Каждый робот начинает движение в тот момент, когда до него доходит стартовый сигнал от активного устройства. Сигнал распространяется со скоростью 1 метр в секунду. Если ближайшее к $i$-му роботу активное устройство находится на $j$-й дорожке, то расстояние между ними составляет $x_i = |i-j|$ метров. Этот робот начнёт движение через $x_i$ секунд после старта, преодолеет дистанцию за $a_i$ секунд, и финиширует через $f_i = a_i + x_i$ секунд после старта робомарафона.

Пусть $k_i$ — количество роботов, которые финишировали строго раньше $i$-го робота. Место $i$-го робота по итогам робомарафона равно $k_i + 1$. Если несколько роботов финишируют одновременно, а перед ними финишировали $k$ роботов, то считается, что все они заняли $(k+1)$-е место.

Рассмотрим пример. Пусть $n = 3$, роботы преодолевают дистанцию за $a_1 = 2$, $a_2 = 3$ и $a_3 = 5$ миллисекунд, а активным являлось только сигнальное устройство у третьего робота. Тогда первый робот начнет движение через 2 секунды после начала забега, $f_1 = 4$. Второй робот начнет движение через 1 секунду, $f_2 = 4$. Третий робот начнет движение в момент старта, $f_3 = 5$. По итогам забега первый и второй робот делят первое место, третий робот занимает третье место. Если же, например, сработают все три сигнальных устройства, роботы финишируеют через $f_1 = 2$, $f_2 = 3$, $f_3 = 5$, секунд, соответственно. Первый робот займет первое место, второй робот займет второе место, а третий робот — третье место.

Как видно из примера, место, которое займет робот, зависит от того, какие сигнальные устройства являются активными. Необходимо обрабатывать два типа запросов:

1. для каждого робота определить минимальное место, которое он может занять;
2. для каждого робота определить максимальное место, которое он может занять.

Требуется написать программу, которая по типу запроса и информации о времени прохождения дистанции каждым роботом определяет для каждого робота минимальное или максимальное место, которое он может занять в марафоне.

## Формат входных данных

В первой строке входных данных находятся два целых числа: $n$ — количество роботов ($1 \leqslant n \leqslant 200\,000$), и $p$ — тип запроса. Значение $p = 1$ означает, что для каждого робота необходимо определить минимальное место, которое он может занять, значение $p = 2$ означает, что для каждого робота необходимо определить максимальное место, которое он может занять.

Во второй строке находятся $n$ целых чисел $a_1, a_2, \ldots, a_n$ — время, за которое роботы преодолевают дистанцию ($0 < a_i \leqslant 10^9$).

## Формат выходных данных

Требуется вывести $n$ целых чисел, $i$-е из которых, в зависимости от типа запроса, должно задавать минимальное или максимальное место, которое может занять $i$-й робот.

## Примеры

| стандартный ввод | стандартный вывод |
|---|---|
| 5 1<br>8 5 5 7 7 | 3<br>1<br>1<br>2<br>1 |
| 5 2<br>8 5 5 7 7 | 5<br>3<br>2<br>4<br>5 |

## Система оценивания

Группа тестов для подзадачи 3 включает 30 тестов. Каждый из этих тестов оценивается независимо в 1 балл.

Группа тестов для подзадачи 4 включает 50 тестов. Каждый из этих тестов оценивается независимо в 1 балл.

| Подзадача | Баллы | Ограничения | | Необходимые подзадачи | Результаты во время тура |
|---|---|---|---|---|---|
| | | $n$ | $p$ | | |
| 1 | 10 | $n \leqslant 20$ | $p = 1$ | | Первая ошибка |
| 2 | 10 | $n \leqslant 20$ | $p = 2$ | | Первая ошибка |
| 3 | до 30 | $n \leqslant 200\,000$ | $p = 1$ | 1 | Баллы |
| 4 | до 50 | $n \leqslant 200\,000$ | $p = 2$ | 2 | Баллы |

Значения $n$ для всех тестов в подзадаче 3 приведены в следующей таблице.

| Тест | $n$ | Тест | $n$ | Тест | $n$ | Тест | $n$ | Тест | $n$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | $n = 100$ | 7 | $n = 2000$ | 13 | $n = 30000$ | 19 | $n = 90000$ | 25 | $n = 150000$ |
| 2 | $n = 250$ | 8 | $n = 2500$ | 14 | $n = 40000$ | 20 | $n = 99999$ | 26 | $n = 170000$ |
| 3 | $n = 500$ | 9 | $n = 4999$ | 15 | $n = 50000$ | 21 | $n = 110000$ | 27 | $n = 190000$ |
| 4 | $n = 750$ | 10 | $n = 10000$ | 16 | $n = 60000$ | 22 | $n = 120000$ | 28 | $n = 210000$ |
| 5 | $n = 1000$ | 11 | $n = 15000$ | 17 | $n = 70000$ | 23 | $n = 130000$ | 29 | $n = 230000$ |
| 6 | $n = 1500$ | 12 | $n = 20000$ | 18 | $n = 80000$ | 24 | $n = 140000$ | 30 | $n = 250000$ |

Значения $n$ для всех тестов в подзадаче 4 приведены в следующей таблице.

| Тест | $n$ | Тест | $n$ | Тест | $n$ | Тест | $n$ | Тест | $n$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | $n = 50$ | 11 | $n = 1500$ | 21 | $n = 10000$ | 31 | $n = 60000$ | 41 | $n = 160000$ |
| 2 | $n = 100$ | 12 | $n = 1999$ | 22 | $n = 12500$ | 32 | $n = 70000$ | 42 | $n = 170000$ |
| 3 | $n = 150$ | 13 | $n = 2500$ | 23 | $n = 15000$ | 33 | $n = 80000$ | 43 | $n = 180000$ |
| 4 | $n = 200$ | 14 | $n = 3000$ | 24 | $n = 20000$ | 34 | $n = 90000$ | 44 | $n = 190000$ |
| 5 | $n = 300$ | 15 | $n = 4000$ | 25 | $n = 25000$ | 35 | $n = 99999$ | 45 | $n = 200000$ |
| 6 | $n = 400$ | 16 | $n = 4999$ | 26 | $n = 30000$ | 36 | $n = 109999$ | 46 | $n = 210000$ |
| 7 | $n = 500$ | 17 | $n = 6000$ | 27 | $n = 35000$ | 37 | $n = 120000$ | 47 | $n = 220000$ |
| 8 | $n = 750$ | 18 | $n = 7000$ | 28 | $n = 40000$ | 38 | $n = 130000$ | 48 | $n = 230000$ |
| 9 | $n = 1000$ | 19 | $n = 8000$ | 29 | $n = 45000$ | 39 | $n = 140000$ | 49 | $n = 240000$ |
| 10 | $n = 1250$ | 20 | $n = 9000$ | 30 | $n = 50000$ | 40 | $n = 150000$ | 50 | $n = 250000$ |

# Задача 8. Сложение без переносов

Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

Операция побитового «или» для набора целых положительных чисел, записанных в двоичной системе счисления, устроена следующим образом. Результатом её применения является число, в двоичной записи которого единица устанавливается в тех разрядах, в которых содержится единица хотя бы у одного числа из набора.

В редких случаях побитовое «или» можно использовать для сложения целых положительных чисел, записанных в двоичной системе счисления. Сумма набора чисел равна их побитовому «или», если для каждого разряда имеется не более одного числа из этого набора, у которого в этом разряде находится единица. Такие наборы чисел назовём *красивыми*.

Задан набор целых положительных чисел $a_1, a_2, \ldots, a_n$. Необходимо построить красивый набор целых положительных чисел $b_1, b_2, \ldots, b_n$, чтобы для всех $i$ от 1 до $n$ выполнялось условие $b_i \geqslant a_i$, а сумма $b_1 + b_2 + \ldots + b_n$ была минимальна.

Требуется написать программу, которая по двоичной записи чисел $a_1, a_2, \ldots, a_n$ определяет двоичную запись минимального значения суммы искомого красивого набора $b_1, b_2, \ldots, b_n$.

## Формат входных данных

В первой строке записано целое число $n$ — количество чисел в наборе ($2 \leqslant n \leqslant 300\,000$).

Следующие $n$ строк содержат двоичную запись целых положительных чисел $a_i$, по одному в строке. Числа не содержат ведущих нулей, и суммарная длина их двоичных записей не превосходит 300 000.

## Формат выходных данных

Требуется вывести двоичную запись минимальной суммы искомого красивого набора $b_1, b_2, \ldots, b_n$. Ответ необходимо вывести без ведущих нулей.

## Примеры

| стандартный ввод | стандартный вывод |
|---|---|
| 2<br>10<br>10 | 110 |
| 2<br>10100<br>1001 | 11101 |
| 3<br>1<br>1<br>110 | 1011 |

## Система оценивания

Обозначим максимальную длину двоичной записи числа во входных данных как $\max L$.

| Подзадача | Баллы | Ограничения | | Необходимые подзадачи | Результаты во время тура |
|---|---|---|---|---|---|
| | | $n$ | $\max L$ | | |
| 1 | 4 | $n = 2$ | $\max L \leqslant 10$ | | Первая ошибка |
| 2 | 2 | $n = 2$ | $\max L \leqslant 20$ | 1 | Первая ошибка |
| 3 | 2 | $n = 2$ | $\max L \leqslant 100$ | 1, 2 | Первая ошибка |
| 4 | 2 | $n = 2$ | $\max L \leqslant 1000$ | 1 – 3 | Первая ошибка |
| 5 | 2 | $n = 2$ | $\max L \leqslant 300\,000$ | 1 – 4 | Первая ошибка |
| В подзадачах 6 – 8 дополнительно $a_i = 2^{k_i}$ для некоторого $k_i$ | | | | | |
| 6 | 4 | $n \leqslant 100$ | $\max L \leqslant 100$ | | Первая ошибка |
| 7 | 4 | $n \leqslant 1000$ | $\max L \leqslant 1000$ | 6 | Первая ошибка |
| 8 | 4 | $n \leqslant 300\,000$ | $\max L \leqslant 300\,000$ | 6, 7 | Первая ошибка |
| 9 | 4 | $n \leqslant 5$ | $\max L \leqslant 5$ | У | Первая ошибка |
| 10 | 4 | $n \leqslant 5$ | $\max L \leqslant 1\,000$ | У, 1 – 4, 9 | Первая ошибка |
| 11 | 4 | $n \leqslant 1\,000$ | $\max L \leqslant 5$ | У, 9 | Первая ошибка |
| 12 | 4 | $n \leqslant 10$ | $\max L \leqslant 10$ | У, 1, 9 | Первая ошибка |
| 13 | 4 | $n \leqslant 50$ | $\max L \leqslant 50$ | У, 1, 2, 9, 12 | Первая ошибка |
| 14 | 7 | $n \leqslant 100$ | $\max L \leqslant 100$ | У, 1 – 3, 6, 9, 12, 13 | Первая ошибка |
| 15 | 7 | $n \leqslant 300$ | $\max L \leqslant 300$ | У, 1 – 3, 6, 9, 12 – 14 | Первая ошибка |
| 16 | 8 | $n \leqslant 1000$ | $\max L \leqslant 1000$ | У, 1 – 4, 6, 7, 9 – 15 | Первая ошибка |
| 17 | 8 | $n \leqslant 3000$ | $\max L \leqslant 3000$ | У, 1 – 4, 6, 7, 9 – 16 | Первая ошибка |
| 18 | 6 | $n \leqslant 10\,000$ | $\max L \leqslant 10\,000$ | У, 1 – 4, 6, 7, 9 – 17 | Первая ошибка |
| 19 | 7 | $n \leqslant 30\,000$ | $\max L \leqslant 30\,000$ | У, 1 – 4, 6, 7, 9 – 18 | Первая ошибка |
| 20 | 7 | $n \leqslant 100\,000$ | $\max L \leqslant 100\,000$ | У, 1 – 4, 6, 7, 9 – 19 | Первая ошибка |
| 21 | 6 | $n \leqslant 300\,000$ | $\max L \leqslant 300\,000$ | У, 1 – 20 | Первая ошибка |