Задача A. Сортировка слиянием с приколом

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это задача требует от вас написать сортировку слиянием для за заданного массива $A = \langle a_1, a_2, \ldots, a_n \rangle$.

Разумеется, все не так просто. Мы хотим, чтобы вы не смогли сдать какой-нибудь $a.sort()$. А значит, мы попросим от вас посчитать количество инверсий в массиве. Для этого вам потребуется модифицировать стадию слияния массивов — попробуйте понять, что происходит с числом инверсий при слиянии двух массивов. Мы подскажем, что в отсортированном массиве 0 инверсий, а операция merge принимает два отсортированных массива и сливает их в один большой отсортированный массив.

Количество инверсий — это количество пар (i, j) таких, что $i < j$ и $a_i > a_j$. Обратите внимание на то, что ответ может не влезать в 32-битный тип данных, если вы пишете не на python.

Формат входных данных

Первая строка входного файла содержит натуральное число n ($1 \leqslant n \leqslant 100\,000$) — количество элементов массива. Вторая строка содержит n попарно различных элементов массива A — целых неотрицательных чисел, не превосходящих 10^9.

Формат выходных данных

В первой строке выведите одно число — количество инверсий в массиве.

Во второй строке выведите отсортированный массив, элементы выводите через пробел.

Примеры

стандартный ввод	стандартный вывод
5 6 11 18 28 31	0 6 11 18 28 31
4 1 3 2 4	1 1 2 3 4

Задача B. Anti-qsort test

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Рассмотрим алгоритм быстрой сортировки, с выбором в качестве барьерного элемента среднего элемента на отрезке (`q = a[(l + r) / 2]`):

```
void qsort(vector<int> & a, int left, int right)
// Сортировка a[left...right] включительно
{
    if (right <= left)
        return;
    int q = a[(l + r) / 2];
    int i = left;
    int j = right;
    while (i <= j) {
        while (a[i] < q)
            ++i;
        while (q < a[j])
            --j;
        if (i <= j) {
            swap(a[i], a[j]);
            ++i;
            --j;
        }
    }
    qsort(a, left, j);
    qsort(a, i, right);
}
```

По данному числу n составьте тест, являющийся перестановкой чисел от 1 до n, на котором этот алгоритм выполняет наибольшее число сравнений (подсчитываются сравнения `a[i] < q` и `q < a[j]`.

Формат входных данных

Программа получает на вход одно целое число n, $1 \leqslant n \leqslant 70\,000$.

Формат выходных данных

Программа должна вывести перестановку чисел от 1 до n, на которой данная реализация алгоритма быстрой сортировки Хоара будет выполнять наибольшее число сравнений.

Можно вывести любой из возможных ответов.

Пример

стандартный ввод	стандартный вывод
3	1 3 2

Задача C. Число

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Вася написал на длинной полоске бумаги большое число и решил похвастаться своему старшему брату Пете этим достижением. Но только он вышел из комнаты, чтобы позвать брата, как его сестра Катя вбежала в комнату и разрезала полоску бумаги на несколько частей. В результате на каждой части оказалось одна или несколько идущих подряд цифр.

Теперь Вася не может вспомнить, какое именно число он написал. Только помнит, что оно было очень большое. Чтобы утешить младшего брата, Петя решил выяснить, какое максимальное число могло быть написано на полоске бумаги перед разрезанием. Помогите ему!

Формат входных данных

Входной файл содержит одну или более строк, каждая из которых содержит последовательность цифр. Количество строк во входном файле не превышает 10000, каждая строка содержит от 1 до 100 цифр. Гарантируется, что хотя бы в одной строке первая цифра отлична от нуля.

Формат выходных данных

Выведите в выходной файл одну строку — максимальное число, которое могло быть написано на полоске перед разрезанием.

Примеры

стандартный ввод	стандартный вывод
2 20 004 66	66220004
3	3

Задача D. Зеркальный код

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Недавно Петя достад с полки запылившуюся детскую энциклопедию и прочитал там про метод шифрования, который использовал сам Леонардо да Винчи. Метод относитлельно прост — писать зеркально. Ходят слухи, что для этого он просто прикладывал зеркало к краю страницы и писал, смотря в отражение.

Но Петя знает, что есть строки, которым такое шифрование ничего не сделает. Строка, которая одинаково читаются справа налево и слева направо называются палиндромами.

У Пети есть обычная строка, состоящая из больших букв латинского алфавита. Ему стало интересно, можно ли составить из каких-то букв этой строки палиндром. Петя хочет получить палиндром наибольшей длины, а из всех таких — первый в алфавитном порядке.

Так как Пете еще надо дочитать энциклопедию, то он попросил сделать это вас.

Формат входных данных

В первой строке входных данных содержится число N ($1 \leqslant N \leqslant 100000$). Во второй строке задается строка из N больших латинских букв (буквы записаны без пробелов).

Формат выходных данных

В единственной строке выходных данных выдайте искомый палиндром.

Примеры

стандартный ввод	стандартный вывод
3 AAB	ABA
6 QAZQAZ	AQZZQA
6 ABCDEF	A

Задача E. Что? Да! Пузырек

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вам дан код сортировки пузырьком:

```
for iteration in range(n):
    had_swaps = False
    for i in range(n - 1):
        if a[i] > a[i + 1]:
            a[i], a[i + 1] = a[i + 1], a[i]
            had_swaps = True
    if not had_swaps:
        break
```

А еще дан массив размера n, состоящий из нулей. Все 0 по очереди превращаются в 1, в порядке, заданном в тесте. От вас требуется после каждой замены говорить, сколько внешних циклов сделаем алгоритм сортировки (до вызова break), если его запустить на массиве.

Заметьте, что вы не должны сортировать, вы должны только сказать, сколько итераций понадобилось бы на сортировку.

Формат входных данных

В первой строке задано целое число n ($1 \leqslant n \leqslant 300000$) — количество элементов в массиве.

Следующая строка содержит n целых различных чисел $p_1, p_2, \ldots, p_n$ ($1 \leqslant pi \leqslant n$) — позиции нулей, если смотреть слева направо, которые меняются на 1. Сначала меняется ноль, находящийся на позиции p_1, затем на позиции p_2 и так далее.

Формат выходных данных

Выведите $n+1$ число $a_0, a_1, \ldots, a_n$, где a_0 — количество итераций для упорядочивания последовательности в начале, a_1 — сложность упорядочивания после первой замены и так далее.

Примеры

стандартный ввод	стандартный вывод
4 1 3 4 2	1 2 3 2 1
11 10 8 9 4 6 3 5 1 11 7 2	1 2 3 4 5 6 7 8 9 6 2 1

Замечание

В первом тесте сначала нужно отсортировать $[0, 0, 0, 0]$, это можно сделать за один проход (который поймет, что все отсортировано).

Затем нужно отсортировать $[1, 0, 0, 0]$, это можно сделать за 2 прохода. Первый проход сделает все свапы, а второй проверит, что больше свапы не нужны.

Затем нужно отсортировать $[1, 0, 1, 0]$, на что потребуется 3 прохода.

Затем сортируется $[1, 0, 1, 1]$, на что потребуется 2 прохода - на первую единицу и финальная проверка сортированности.

Для сортировки $[1, 1, 1, 1]$ нужен один проход.