Задача A. Сортировка

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

Отсортируйте массив целых чисел в порядке неубывания, **используя сортировку выбором минимума**.

Формат входных данных

Первая строка входного файла содержит целое число N ($1 \leqslant N \leqslant 1000$), вторая строка — N целых чисел, по модулю не превышающих $2 \cdot 10^9$.

Формат выходных данных

Данные числа следует вывести в порядке неубывания.

Примеры

sort.in	sort.out
5 9 2 7 1 2	1 2 2 7 9

Замечание

Необходимо написать функцию sort, которая принимает список и сортирует его.

При решении этой задачи нельзя пользоваться стандартными функциями и методами `min`, `index`, `sort`, `sorted` и т. д.

Естественно, можно пользоваться функциями `min`, `max`, которые принимают два числа.

Задача B. Ревизия

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

В связи с визитом Императора Палпатина было решено обновить состав дроидов в ангаре 32. Из-за кризиса было решено новых дроидов не закупать, но выкинуть пару старых. Как известно, Палпатин не переносит дроидов с маленькими серийными номерами, так что всё, что требуется — найти среди них двух, у которых серийные номера наименьшие.

Формат входных данных

Первая строка входного файла содержит целое число N — количество дроидов ($2 \leqslant N \leqslant 100\,000$), вторая строка — N целых чисел, по модулю не превышающих $2 \cdot 10^9$ — номера дроидов.

Формат выходных данных

Выведите два числа: первым — наименьший серийный номер дроида (которого поэтому следует утилизировать в первую очередь), а вторым — второй по минимальности.

Примеры

inspection.in	inspection.out
5 49 100 23 -100 157	-100 23
3 1 2419 1	1 1

Замечание

При решении этой задачи нельзя пользоваться стандартными функциями и методами `min`, `index`, `sort`, `sorted` и т. д.

Естественно, можно пользоваться функциями `min`, `max`, которые принимают два числа.

Задача C. Циклический сдвиг

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

Вам требуется написать функцию, которая получает на вход массив и изменяет его таким образом, чтобы на первом месте стоял последний элемент, на втором — первый, на третьем — второй и т. д.

Формат входных данных

В первой строчке находится число N ($1 \leqslant N \leqslant 100\,000$). В следующей строке через пробел перечислены все числа ($1 \leqslant a_i \leqslant 10^9$).

Формат выходных данных

Выведите N чисел — измененный массив.

Примеры

shift.in	shift.out
5 21 13 7 2 6	6 21 13 7 2

Замечание

Программы, в которых отсутствует или не вызывается требуемая процедура, будут проигнорированы.

Задача D. Сортировка вставками

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

Отсортируйте массив целых чисел в порядке невозрастания, **используя сортировку вставками**.

Формат входных данных

Первая строка входного файла содержит целое число N ($1 \leqslant N \leqslant 1000$), вторая строка — N целых чисел, по модулю не превышающих $2 \cdot 10^9$.

Формат выходных данных

Данные числа следует вывести в порядке невозрастания.

Примеры

sort.in	sort.out
5 9 2 7 1 2	9 7 2 2 1
5 5 4 3 2 1	5 4 3 2 1

Замечание

В этой задаче нельзя пользоваться методом .sort() и функцией sorted(), а также использовать метод .reverse()

Задача E. Сортировка подсчетом

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

Вам дан массив. Требуется его отсортировать.

Формат входных данных

В первой строке входного файла находится одно целое число n — количество элементов массива ($1 \leqslant n \leqslant 200000$). Во второй строчке находятся n натуральных чисел — элементы массива. Все элементы массива не превосходят 10^4.

Формат выходных данных

В единственную строку выходного файла выведите отсортированный массив.

Примеры

countsort.in	countsort.out
3 1 2 3	1 2 3
3 3 2 1	1 2 3

Задача F. Имперский марш

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

На этот раз Император нагрянул с ревизией не в какой-то там ангар, а в казармы 501-го легиона имперских штурмовиков. В связи с этим каждого штурмовика постригли «под ежика». Несмотря на развитие нанотехнологий, постригли плохо — в результате из-за различной длины волос штурмовики могут отличаться друг от друга по росту, но незначительно — разница не превышает 137 нанометров. Ваша задача — выстроить штурмовиков по росту.

Формат входных данных

Первая строка входного файла содержит целое число N — количество штурмовиков ($1 \leqslant N \leqslant 100\,000$), вторая строка N — натуральных чисел, не превышающих $2 \cdot 10^9$ каждое — рост штурмовика в нанометрах. Никакие два роста не различаются более, чем на 137 нм.

Формат выходных данных

Выведите роста штурмовиков в порядке неубывания.

Примеры

march.in	march.out
5 12 1 2 1 13	1 1 2 12 13
1 1000000000	1000000000

Замечание

При решении этой задачи нельзя пользоваться стандартными функциями и методами `min`, `index`, `sort`, `sorted` и т. д.

Естественно, можно пользоваться функциями `min`, `max`, которые принимают два числа.