

Задача 1. Быстрая сортировка

Источник: базовая
Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 2 секунды
Ограничение по памяти: разумное

Отсортировать заданную последовательность целых чисел с помощью реализованного вами алгоритма **быстрой сортировки**.

Формат входных данных

В первой строке входного файла записано целое число N — длина последовательности ($1 \leq N \leq 10^6$).

В следующей строке через пробел записано N целых чисел. Все числа по модулю не превосходят 10^6 .

Формат выходных данных

В выходной файл необходимо вывести заданную последовательность в отсортированном виде. Числа выводить через пробел в одну строку.

Пример

input.txt	output.txt
5 12 5 1 -3 4	-3 1 4 5 12

Задача 2. Сортировка Шелла

Источник: базовая
Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 2 секунды
Ограничение по памяти: разумное

Отсортировать заданную последовательность целых чисел с помощью реализованного вами алгоритма **сортировки Шелла**.

Формат входных данных

В первой строке входного файла записано целое число N — длина последовательности ($1 \leq N \leq 10^6$).

В следующей строке через пробел записано N целых чисел. Все числа по модулю не превосходят 10^6 .

Формат выходных данных

В выходной файл необходимо вывести заданную последовательность в отсортированном виде. Числа выводить через пробел в одну строку.

Пример

input.txt	output.txt
5 12 5 1 -3 4	-3 1 4 5 12

Задача 3. Пирамидальная сортировка

Источник:	базовая
Имя входного файла:	<code>input.txt</code>
Имя выходного файла:	<code>output.txt</code>
Ограничение по времени:	2 секунды
Ограничение по памяти:	разумное

Отсортировать заданную последовательность целых чисел с помощью реализованного вами алгоритма **пирамидальной сортировки**.

Формат входных данных

В первой строке входного файла записано целое число N — длина последовательности ($1 \leq N \leq 10^6$).

В следующей строке через пробел записано N целых чисел. Все числа по модулю не превосходят 10^6 .

Формат выходных данных

В выходной файл необходимо вывести заданную последовательность в отсортированном виде. Числа выводить через пробел в одну строку.

Пример

input.txt	output.txt
5 12 5 1 -3 4	-3 1 4 5 12

Задача 4. Удаление дубликатов

Источник: основная
Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 1 секунда*
Ограничение по памяти: разумное

Дан массив A , в котором содержится n целых чисел. Нужно удалить из него дубликаты (т.е. повторы чисел), так чтобы в массиве каждое имеющееся в нём значение встречалось ровно один раз.

Формат входных данных

В первой строке записано целое число n — сколько чисел в массиве ($1 \leq N \leq 300\,000$). В остальных n строках записаны сами эти числа. Все числа по модулю не превышают 10^9 .

Формат выходных данных

В первой строке нужно вывести целое число k — сколько различных чисел в массиве A . В оставшихся k строках нужно вывести сами эти различные числа в любом порядке.

Пример

input.txt	output.txt
10	5
1	1
1	3
-2	-2
4	0
3	4
0	
0	
0	
0	
-2	

Комментарий

Подсказка: подумайте, как решить задачу за $O(N)$, если массив A упорядочен по возрастанию.

Задача 5. Сумма минимумов

Источник:	основная
Имя входного файла:	<code>input.txt</code>
Имя выходного файла:	<code>output.txt</code>
Ограничение по времени:	1 секунда*
Ограничение по памяти:	разумное

Дан массив A , в котором содержится n целых чисел. Нужно перебрать все пары чисел A_i и A_j в этом массиве, для каждой пары найти минимум $\min(A_i, A_j)$ и сложить вместе все эти минимумы.

Более формально, требуется найти сумму:

$$S = \sum_{i < j} \min(A_i, A_j) = \sum_{i=0}^{n-1} \sum_{j=i+1}^{n-1} \min(A_i, A_j)$$

Формат входных данных

В первой строке записано целое число n — сколько чисел в массиве ($1 \leq n \leq 300\,000$). В остальных n строках записаны сами эти числа в том порядке, в котором они идут в массиве A . Все числа по модулю не превышают 10^9 .

Формат выходных данных

Нужно вывести одно целое число — искомую сумму минимумов S .

Осторожно: сумма S может быть довольно большой. Оцените максимально возможное значение S и выберите подходящий целочисленный тип.

Пример

<code>input.txt</code>	<code>output.txt</code>
5 1 4 3 5 6	26

Комментарий

Подсказка: подумайте, как решить задачу за $O(N)$, если массив A упорядочен по возрастанию.

Задача 6. Разница множеств

Источник: основная
Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 1 секунда
Ограничение по памяти: разумное

Дано два массива целых чисел A и B .

Требуется найти все такие значения элементов массива A , которых нет среди элементов массива B .

Замечание: В задаче необходимо использовать функцию `qsort` из стандартной библиотеки языка C.

Формат входных данных

В первой строке записано целое число N ($1 \leq N \leq 10^5$) — количество элементов массива A .

Во второй строке через пробел записано N целых чисел, каждое из которых не превосходит 10^9 по абсолютной величине — элементы массива A .

В следующих двух строках в аналогичном формате записаны элементы массива B .

Формат выходных данных

В первой строке выведите одно целое число — количество значений, удовлетворяющих описанному условию.

Во второй строке выведите все такие значения в порядке возрастания.

Примеры

input.txt	output.txt
7 1 2 3 3 6 8 8	3 2 6 8
4 1 3 7 9	
3 1 2 3 3 3 2 1	0

Задача 7. lower bound

Источник:	основная
Имя входного файла:	<code>input.txt</code>
Имя выходного файла:	<code>output.txt</code>
Ограничение по времени:	2 секунды
Ограничение по памяти:	разумное

Функция `bsearch` из стандартной библиотеки языка C обладает большим недостатком: если в массиве нет того элемента, который мы ищем, то функция не возвращает абсолютно никакой информации. Это означает, что в некоторых случаях она бесполезна, например при решении задачи «Поиск ближайшего».

Гораздо полезнее функция `lower_bound` из стандартной библиотеки языка C++. Эта функция запускается для отсортированного массива A и элемента X , и возвращает номер первого элемента в массиве, который больше или равен X . Если такого элемента нет, то функция возвращает длину массива N . Иными словами, функция возвращает, сколько элементов в массиве A строго меньше заданного X .

В данной задаче вам предлагается реализовать `lower_bound` на языке C аналогично тому, как реализована функция `bsearch`. Это означает, что должны выполняться следующие требования:

1. Функция должна быть применима к массиву элементов любого типа. Это означает, что она должна принимать размер одного элемента в байтах и нетипизированные указатели, аналогично `qsort` и `bsearch`.
2. Функция должна поддерживать задание критерия сравнения для элементов. Это означает, что она должна принимать компаратор в виде указателя на функцию (можно без контекста).
3. Функция должна брать информацию только из своих параметров/аргументов. Иными словами, внутри неё нельзя обращаться к глобальным и статическим переменным.

Данную функцию нужно применить к двум заданным массивам: один состоит из целых чисел, а другой — из строк. Обратите внимание, что изначально массивы не отсортированы — вам следует применить к ним `qsort`.

Формат входных данных

Входные данные задаются в таком порядке: сначала массив целых чисел, затем массив строк, затем запросы для массива целых чисел, и, наконец, запросы для массива строк.

В первом блоке записано одно целое число N_1 — длина массива целых чисел ($1 \leq N_1 \leq 10^5$). Далее записаны элементы этого массива: N_1 целых чисел, каждое по абсолютной величине не превышает 10^{15} .

Во втором блоке записано одно целое число N_2 — длина массива строк ($1 \leq N_2 \leq 10^5$). Далее записаны элементы этого массива: N_2 непустых строк из маленьких букв латинского алфавита, длиной не более 31 символа каждая.

В третьем блоке записано целое число Q_1 — количество запросов для массива целых чисел ($1 \leq Q_1 \leq 10^5$). В остальных Q_1 строках записаны целые числа, определяющие запросы на поиск, каждое число не превышает 10^{15} по абсолютной величине.

В четвёртом блоке записано целое число Q_2 — количество запросов для массива строк ($1 \leq Q_2 \leq 10^5$). В остальных Q_2 строках записаны строки-запросы, состоящие из маленьких букв латинского алфавита, длиной от 1 до 31 символа.

Формат выходных данных

В выходные данные нужно вывести сначала результаты применения `lower_bound` для запросов на массиве целых чисел, а затем результаты применения для запросов на массиве строк. Первый блок результатов должен содержать Q_1 целых чисел в диапазоне от 0 до N_1 , каждое число в отдельной строке. Второй блок результатов должен содержать Q_2 целых чисел в диапазоне от 0 до N_2 , каждое число в отдельной строке.

Пример

input.txt	output.txt
5	2
-1000000000	2
3000000000	4
5000000000	2
-1000000000	4
3000000000	4
5	5
a	0
hello	0
a	3
ba	5
a	
8	
3000000000	
2999999999	
3000000001	
0	
5000000000	
4000000000	
7000000000	
-7000000000	
3	
a	
b	
hi	

Пояснение к примеру

Массив целых чисел в отсортированном виде выглядит как: $-G$, $-G$, $3G$, $3G$, $5G$ (для краткости обозначим $G = 10^9$). Первый запрос в точности равен $3G$, он впервые встречается под индексом 2 в массиве. Второй запрос чуть меньше $3G$, и в массиве отсутствует, так что нужно вернуть индекс первого элемента больше него, а это 2. Третий запрос чуть больше $3G$, и в массиве также отсутствует. В массиве всего 4 числа меньше запрашиваемого, так что ответ равен 4. Последние два запроса показывают, что нужно возвращать, когда запрашиваемое число больше или меньше всего массива.

Массив строк в отсортированном виде выглядит так: a, a, a, ba, hello. Для запроса a ответ равен 0, т.к. элементов меньше в массиве нет. Для запроса b ответ равен 3, т.к. все три строки a меньше него, а строка ba больше него. Запрос hi больше всего массива, так что ответ равен длине массива.

Задача 8. Выпуклый минимум

Источник: дополнительная
Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 1 секунда*
Ограничение по памяти: разумное

Массив чисел $A_0, A_1, A_2, \dots, A_{n-1}$ называется выпуклым вверх, если:

$$\forall i < k < j: \quad A_k < \frac{(j-k)A_i + (k-i)A_j}{(j-i)}$$

Дан выпуклый вверх массив A и коэффициент C . Требуется найти индекс элемента массива, на котором достигается минимум линейной функции:

$$\operatorname{argmin}_{i=0}^{n-1} (A_i + C \cdot i) = ?$$

Если минимальное значение достигается на нескольких элементах массива, нужно найти номер первого такого элемента.

Формат входных данных

В первой строке записано одно целое число n — размер выпуклого массива ($1 \leq n \leq 10^5$). Далее записаны элементы массива A_i (n целых чисел, $|A_i| \leq 10^{15}$). Затем записано целое число q — количество запросов, которые нужно обработать ($1 \leq q \leq 10^5$). В остальных q строках записаны целые числа C_j , определяющие значения коэффициента линейной функции ($|C_j| \leq 10^9$).

Формат выходных данных

Требуется вывести q целых чисел: для каждого коэффициента C_j , записанного во входных данных, нужно вывести номер i первого элемента A_i , на котором достигается минимум $(A_i + C \cdot i)$ при $C = C_j$.

Пример

input.txt	output.txt
10	8
9 4 0 -2 -2 -1 1 4 8 20	3
8	5
-5	3
1	0
-2	2
0	2
6	1
3	
2	
4	

Пояснение к примеру

Рассмотрим коэффициент $C_2 = -2$. Выпишем значение соответствующей функции для всех элементов:

$$i=0: 9 - 2 * 0 = 9$$

$$i=1: 4 - 2 * 1 = 2$$

$$i=2: 0 - 2 * 2 = -4$$

$$i=3: -2 - 2 * 3 = -8$$

$$i=4: -2 - 2 * 4 = -10$$

$$i=5: -1 - 2 * 5 = -11$$

$$i=6: 1 - 2 * 6 = -11$$

$$i=7: 4 - 2 * 7 = -10$$

$$i=8: 8 - 2 * 8 = -8$$

$$i=9: 20 - 2 * 9 = 2$$

Минимум достигается на двух элементах $i = 5$ и $i = 6$, и ответом является меньший номер $i = 5$.

Комментарий

Представьте себе, как бы вы решали задачу, если бы вместо массива A была дана гладкая функция $A(x)$, и нужно было бы найти минимум функции $(A(x) + Cx)$. Задача с массивом решается точно так же, нужно лишь найти дискретный аналог для понятия производной.

Задача 9. Булевы формулы

Источник:	повышенной сложности
Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Задано две булевы формулы, первая является полиномом Жегалкина, а вторая представлена в конъюнктивной нормальной форме. Требуется определить, эквивалентны они или нет.

Формат входных данных

В первой строке записано целое число T — количество тестов в файле ($1 \leq T \leq 100$). Далее описывается T тестов, по две строки на тест. В первой строке теста записан полином Жегалкина, а во второй — формула в КНФ.

Каждая формула записывается без пробелов и состоит только из маленьких латинских букв от a до j (первые 10 букв) и следующих символов:

- '&' (ASCII 38) — конъюнкция,
- '|' (ASCII 124) — дизъюнкция (только в КНФ),
- '!' (ASCII 33) — отрицание (только в КНФ),
- '+' (ASCII 43) — сложение по модулю два (только в полиноме Жегалкина),
- '1' (ASCII 49) — истина (только в полиноме Жегалкина).

Буквы обозначают различные булевы переменные.

Обе формулы синтаксически корректны с точки зрения алгебры логики. В формулах отсутствуют скобки, так что в КНФ надо считать, что дизъюнкция имеет больший приоритет, чем конъюнкция. В КНФ перед каждой переменной либо стоит один символ отрицания, либо отрицания нет (гарантируется, что кратных символов отрицания нет). Гарантируется, что константа истины 1 не умножается ни на какую переменную, а встречается в сумме только сама по себе.

Длина каждой формулы не превышает 5 000 символов.

Формат выходных данных

Требуется вывести T строк, в каждой строке ответ на соответствующий тест. Если формулы эквивалентны, нужно написать `Equivalent`, а иначе — `Not equivalent`.

Пример

input.txt	output.txt
2	Not equivalent
a&b&c+1	Equivalent
!a b&a !b&c	
a+b+c	
a b c&!a !b c&!a b !c&a !b !c	

Пояснение к примеру

В первом тесте заданы неэквивалентные формулы: $(a \wedge b \wedge c) \oplus 1$ и $(\bar{a} \vee b) \wedge (a \vee \bar{b}) \wedge c$. Во втором тесте заданы эквивалентные формулы: $a \oplus b \oplus c$ и $(a \vee b \vee c) \wedge (\bar{a} \vee \bar{b} \vee c) \wedge (\bar{a} \vee b \vee \bar{c}) \wedge (a \vee \bar{b} \vee \bar{c})$.

Задача 10. Маленькая сортирующая машина

Источник:	космической сложности
Имя входного файла:	<code>input.txt</code>
Имя выходного файла:	<code>output.txt</code>
Ограничение по времени:	1 секунда*
Ограничение по памяти:	разумное

Вам предлагается испытать себя в оптимизации сортировки массива маленького размера.

В каждом тесте имеется массив из N элементов, у каждого элемента есть ключ и значение. И ключ, и значение являются беззнаковыми четырёхбайтовыми целыми числами.

Изначально во входном файле заданы только значения всех элементов массива. Далее нужно выполнить R раундов. На каждом раунде нужно:

1. Сгенерировать и записать N случайных чисел в ключи элементов массива.
2. Отсортировать элементы массива в порядке возрастания ключа.

В результате каждого раунда значения в массиве переставляются в некотором порядке, который зависит от генератора псевдослучайных чисел.

В данной задаче нужно использовать генератор псевдослучайных чисел xorwow. Исходный код этого генератора:

```
uint32_t xorwow(uint32_t state[5]) {
    uint32_t s, t = state[3];
    t ^= t >> 2;
    t ^= t << 1;
    state[3] = state[2]; state[2] = state[1]; state[1] = s = state[0];
    t ^= s;
    t ^= s << 4;
    state[0] = t;
    return t + (state[4] += 362437);
}
```

Начальное состояние генератора (пять чисел в `state`) задаётся в каждом тесте. В начале каждого раунда ключи генерируются для элементов в их текущем порядке, причём старшие два бита отбрасываются:

```
uint32_t state[5] = {..., ..., ..., ..., ...};
for (int i = 0; i < n; i++)
    elements[i].key = xorwow(state) & 0x3fffffff;
```

В конце теста нужно вывести значения всех элементов массива в их финальном порядке. Обратите внимание, что процесс полностью детерминирован, и только один финальный порядок является правильным.

Формат входных данных

В первой строке записано целое число Q — сколько тестов записано в файле ($1 \leq Q \leq 1000$). Далее описано Q тестов.

Описание теста начинается со строки с двумя целыми числами: N — размер массива и R — сколько раундов сортировки нужно выполнить ($1 \leq N \leq 64$, $0 \leq R$). Во второй строке теста записано пять шестнадцатеричных чисел, по восемь цифр в каждом — начальное содержимое массива `state` генератора xorwow. В третьей строке записано N целых чисел — значения элементов массива в том порядке, в котором они расположены изначально.

Гарантируется, что во всех раундах всех тестов все ключи будут различными. Суммарное количество раундов R по всем тестам в файле не превышает 750 000.

Формат выходных данных

Для каждого из Q тестов нужно вывести ровно одну строку. В этой строке должно быть N целых чисел: значения элементов массива в их финальном порядке после всех раундов.

Пример

input.txt
5 15 0 b1c6114b f18c80b8 059cace1 24e9297b 5cab5281 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 15 1 b1c6114b f18c80b8 059cace1 24e9297b 5cab5281 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 15 2 b1c6114b f18c80b8 059cace1 24e9297b 5cab5281 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 15 3 b1c6114b f18c80b8 059cace1 24e9297b 5cab5281 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 10 7 4c373cdb 0102026b a8b5ef27 370796de 5840f014 135 12 13 11 10 17 10 7 1 5
output.txt
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 2 6 5 10 3 12 8 14 13 11 4 9 7 15 1 1 11 12 4 5 3 6 14 2 10 8 15 9 13 7 5 12 1 11 8 13 14 15 3 10 2 9 7 6 4 12 1 11 13 135 10 7 10 17 5

Комментарий

В данной задаче бесовестно жёсткое ограничение по времени. Скорее всего никто не сможет уложиться в TL — тогда задача перейдёт в разряд соревнования: кто сможет написать самое быстрое решение.

Внимание: не пытайтесь применять многопоточность! В nsuts замеряется суммарное процессорное время по всем потокам, поэтому многопоточность не поможет.