Задача A. Хип ли?

Имя входного файла: `isheap.in`
Имя выходного файла: `isheap.out`

Структуру данных Heap можно реализовать на основе массива.

Для этого должно выполнятся *основное свойство Heap'a*, которое заключается в следующем. Для каждого $0 \leqslant i < n$ выполняются следующие условия:

- Если $2i + 1 < n$, то $a[i] \leqslant a[2i + 1]$
- Если $2i + 2 < n$, то $a[i] \leqslant a[2i + 2]$

Дан массив целых чисел. Определите является ли он Heap'ом.

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \leqslant n \leqslant 10^5$). Вторая строка содержит n целых чисел по модулю не превосходящих $2 \cdot 10^9$.

Формат выходного файла

Выведите «`YES`», если массив является Heap'ом и «`NO`» в противном случае.

Пример

isheap.in	isheap.out
5 1 0 1 2 0	NO
5 1 3 2 5 4	YES

Задача B. Приоритетная очередь

Имя входного файла: `priorityqueue.in`
Имя выходного файла: `priorityqueue.out`

Реализуйте приоритетную очередь. Ваша очередь должна поддерживать следующие операции: добавить элемент, извлечь минимальный элемент, уменьшить элемент, добавленный во время одной из операций.

Все операции нумеруются по порядку, начиная с 1.

Формат входного файла

Входной файл содержит описание операций со очередью. В очередь помещаются и извлекаются только целые числа, не превышающие по модулю 10^9.

Формат выходного файла

Выведите последовательно результат выполнения всех операций `extract-min`. Если перед очередной операции `extract-min` очередь пуста, выведите вместо числа звездочку.

Пример

priorityqueue.in	priorityqueue.out
push 3 push 4 push 2 extract-min decrease-key 2 1 extract-min extract-min extract-min	2 1 3 *

Задача C. Сортировка

Имя входного файла: `sort.in`
Имя выходного файла: `sort.out`

Дан массив целых чисел. Ваша задача — отсортировать его в порядке неубывания.

Для того, чтобы узнать, какую сортировку вам нужно реализовать, возьмите остаток от деления вашего номера на 3.

- 0 — Сортировка слиянием.
- 1 — Сортировка кучей.
- 2 — Быстрая сортировка.

Формат входного файла

В первой строке входного файла содержится число n ($1 \le n \le 100000$) — количество элементов в массиве. Во второй строке находятся n целых чисел, по модулю не превосходящих 10^9.

Формат выходного файла

В выходной файл надо вывести этот же массив в порядке неубывания, между любыми двумя числами должен стоять ровно один пробел.

Пример

sort.in	sort.out
10 1 8 2 1 4 7 3 2 3 6	1 1 2 2 3 3 4 6 7 8

Задача D. K-ая порядковая статистика

Имя входного файла: `kth.in`
Имя выходного файла: `kth.out`

Дан массив из n элементов. Какое число k-ое в порядке возрастания в этом массиве.

Формат входного файла

В первую строке входного файла содержится два числа n — размер массива и k.($1 \le k \le n \le 3 \cdot 10^7$). Во второй строке находятся числа A, B, C, a_1, a_2 по модулю не превосходящие 10^9. Вы должны получить элементы массива начиная с третьего по формуле: $a_i = A * a_{i-2} + B * a_{i-1} + C$. Все вычисления должны производится в 32 битном знаковом типе, переполнения должны игнорироваться.

Формат выходного файла

Выведите значение k-ое в порядке возрастания число в массиве a.

Пример

kth.in	kth.out
5 3 2 3 5 1 2	13
5 3 200000 300000 5 1 2	2

Во втором примере элементы массива a равны: (1, 2, 800005, −516268571, 1331571109).

Задача E. Цифровая сортировка

Имя входного файла: `radixsort.in`
Имя выходного файла: `radixsort.out`

Дано n строк, выведите их порядок после k фаз цифровой сортировки.

Формат входного файла

В первой строке входного файла содержится число n — количество строк, m — их длина и k – число фаз цифровой сортировки ($1 \le n \le 1000$, $1 \le k \le m \le 1000$). В следующих n строках находятся сами строки.

Формат выходного файла

Выведите строки в порядке в котором они будут после k фаз цифровой сортировки.

Пример

radixsort.in	radixsort.out
3 3 1 bbb aba baa	aba baa bbb
3 3 2 bbb aba baa	baa aba bbb
3 3 3 bbb aba baa	aba baa bbb

Задача F. Анти-QuickSort

Имя входного файла: `antiqs.in`
Имя выходного файла: `antiqs.out`

Для сортировки последовательности чисел широко используется быстрая сортировка — QuickSort. Далее приведена программа, которая сортирует массив a, используя этот алгоритм.

```
def qsort(left, right):
    key = a[(left + right) // 2]
    i = left
    j = right
    while i <= j:
        while a[i] < key:     # first while
            i += 1
        while a[j] > key:     # second while
            j -= 1
        if i <= j:
            a[i], a[j] = a[j], a[i]
            i += 1
            j -= 1
    if left < j:
        qsort(left, j)
    if i < right:
        qsort(i, right)


qsort(0, n - 1)
```

Хотя QuickSort является самой быстрой сортировкой в среднем, существуют тесты, на которых она работает очень долго. Оценивать время работы алгоритма будем количеством сравнений с элементами массива (то есть суммарным количеством сравнений в первом и втором while). Требуется написать программу, генерирующую тест, на котором быстрая сортировка сделает наибольшее число таких сравнений.

Формат входного файла

В первой строке находится единственное число n ($1 \le n \le 70000$).

Формат выходного файла

Вывести перестановку чисел от 1 до n, на которой быстрая сортировка выполнит максимальное число сравнений. Если таких перестановок несколько, вывести любую из них.

Пример

antiqs.in	antiqs.out
3	1 3 2

Задача G. Стильная одежда

Имя входного файла: `style.in`
Имя выходного файла: `style.out`

Глеб обожает шоппинг. Как-то раз он загорелся идеей подобрать себе майку и штаны так, чтобы выглядеть в них максимально стильно. В понимании Глеба стильность одежды тем больше, чем меньше разница в цвете элементов его одежды.

В наличии имеется N ($1 \le N \le 100\,000$) маек и M ($1 \le M \le 100\,000$) штанов, про каждый элемент известен его цвет (целое число от 1 до 10 000 000). Помогите Глебу выбрать одну майку и одни штаны так, чтобы разница в их цвете была как можно меньше.

Формат входного файла

Сначала вводится информация о майках: в первой строке целое число N ($1 \le N \le 100\,000$) и во второй N целых чисел от 1 до 10 000 000 — цвета имеющихся в наличии маек. **Гарантируется**, что номера цветов идут в возрастающем порядке (в частности, цвета никаких двух маек не совпадают).

Далее в том же формате идёт описание штанов: их количество M ($1 \le M \le 100\,000$) и в следующей строке M целых чисел от 1 до 10 000 000 в возрастающем порядке — цвета штанов.

Формат выходного файла

Выведите пару неотрицательных чисел — цвет майки и цвет штанов, которые следует выбрать Глебу. Если вариантов выбора несколько, выведите любой из них.

Примеры

style.in	style.out
2 3 4 3 1 2 3	3 3
2 4 5 3 1 2 3	4 3

Задача H. Стильная одежда 2

Имя входного файла: `style.in`
Имя выходного файла: `style.out`

Глеб решил развить успех и купить более сложный комплект: из кепки, майки, штанов и ботинок. В понимании Глеба стильность комплекта тем больше, чем меньше максимальная разница в цвете элементов его одежды.

В наличии имеется N_1 кепок, N_2 маек, N_3 штанов и N_4 ботинок, про каждый элемент известен его цвет. Помогите Глебу выбрать комплект.

Формат входного файла

Сначала вводится информация о кепках, потом о майках, потом о штанах и в конце о ботинках:. Про каждый тип одежды в первой строке целое число N_i ($1 \le N_i \le 100\,000$) и во второй N_i целых чисел от 1 до 100 000 — цвета имеющихся предметов.

Формат выходного файла

Выведите четыре числа — цвета кепки, майки, штанов и ботинок. Если вариантов выбора несколько, выведите любой из них.

Примеры

style.in	style.out
3 1 2 3 2 1 3 2 3 4 2 2 3	3 3 3 3
1 5 4 3 6 7 10 4 18 3 9 11 1 20	5 7 9 20

Задача I. Поезда

Имя входного файла: `trains.in`
Имя выходного файла: `trains.out`

В связи с участившимся числом аварий на железнодорожной ветке Москва–Саратов, руководство железной дороги решило изменить график движения поездов. Тщательный анализ состояния железнодорожного полотна показал, что оптимальным является следующий график движения поездов с учетом остановок на станциях: сначала поезд идет на протяжении T_1 минут со скоростью V_1 метров в минуту, затем T_2 минут со скоростью V_2 метров в минуту, ..., наконец, T_N минут со скоростью V_N метров в минуту. В течение некоторых интервалов поезд может стоять (скорость равна 0).

По действующей инструкции обеспечения безопасности движения поездов расстояние между локомотивами двух следующих друг за другом поездов должно быть не менее L метров. Определите минимально допустимый интервал в минутах между отправлениями поездов, позволяющий им двигаться по этому графику без опасного сближения.

Формат входного файла

В первых двух строках входного файла содержится два натуральных числа, задающие минимально допустимое расстояние L и количество участков пути N ($100 \leqslant L \leqslant 10\,000$, $1 \leqslant N \leqslant 1000$). Далее следует N пар целых чисел T_i и V_i, задающих график движения поездов ($1 \leqslant T_i \leqslant 1000$, $0 \leqslant V_i \leqslant 1000$).

Формат выходного файла

В выходной файл необходимо вывести искомый интервал между отправлениями поездов в минутах, не менее чем с тремя верными знаками после десятичной точки.

Пример

trains.in	trains.out
1000 4 10 0 30 80 15 0 20 100	27.500