Сдать задание нужно до 30 марта 2019г. (9:00) включительно.

Ссылка на контест: https://contest.yandex.ru/contest/12111/enter/

Задача № 1 (3 балла)

В каждой задаче, где начальными данными является массив вначале вводится количество элементов, а затем и сами элементы массива.

1_1. Даны два массива целых чисел одинаковой длины A[0..n-1] и B[0..n-1]. Необходимо найти первую пару индексов i0 и j0, i0 ≤ j0, такую что A[i0] + B[j0] = max {A[i] + B[j], где 0 <= i < n, 0 <= j < n, i <= j}. Время работы - O(n).

n ≤ 100000.

in	out
4 4 -8 6 0 -10 3 1 1	0 1

1_2. Вычислить площадь выпуклого n-угольника, заданного координатами своих вершин. Вначале вводится количество вершин, затем последовательно целочисленные координаты всех вершин в порядке обхода против часовой стрелки.

n < 1000, координаты < 10000.

Указание. Для вычисления площади n-угольника можно посчитать сумму ориентированных площадей трапеций под каждой стороной многоугольника.

in	out
3 0 1 1 0 2 2	1.5

1_3. Даны два строго возрастающих массива целых чисел A[0..n) и B[0..m) и число k. Найти количество таких пар индексов (i, j), что A[i] + B[j] = k. Время работы O(n + m).

n, m ≤ 100000.

Указание. Обходите массив B от конца к началу.

in	out
4 -5 0 3 18 5 -10 -2 4 7 12 7	3

1_4. “Считалочка”. В круг выстроено N человек, пронумерованных числами от 1 до N. Будем исключать каждого k-ого до тех пор, пока не уцелеет только один человек. (Например, если N=10, k=3, то сначала умрет 3-й, потом 6-й, затем 9-й, затем 2-й, затем 7-й, потом 1-й, потом 8-й, за ним - 5-й, и потом 10-й. Таким образом, уцелеет 4-й.) Необходимо определить номер уцелевшего.

N, k ≤ 10000.

in	out
10 3	4

Задача № 2 (4 балла)

2_1. Дан отсортированный массив целых чисел A[0..n-1] и массив целых чисел B[0..m-1]. Для каждого элемента массива B[i] найдите минимальный индекс k минимального элемента массива A, равного или

превосходящего B[i]: A[k] >= B[i]. Если такого элемента нет, выведите n. Время работы поиска k для каждого элемента B[i]: O(log(k)).
n, m ≤ 10000.
Формат входных данных.
В первой строчке записаны числа n и m. Во второй и третьей массивы A и B соответственно.

in	out
2 1 1 2 2	1
4 3 2 4 5 7 4 6 1	1 3 0

2_2. Дан массив целых чисел A[0..n-1]. Известно, что на интервале [0, m] значения массива строго возрастают, а на интервале [m, n-1] строго убывают. Найти m за O(log m).
2 ≤ n ≤ 10000.

in	out
10 1 2 3 4 5 6 7 6 5 4	6

2_3. Даны два массива неповторяющихся целых чисел, упорядоченные по возрастанию. A[0..n-1] и B[0..m-1]. n >> m. Найдите их пересечение. Требуемое время работы: O(m * log k), где k - позиция элемента B[m-1] в массиве A.. В процессе поиска очередного элемента B[i] в массиве A пользуйтесь результатом поиска элемента B[i-1].
n, k ≤ 10000.

in	out
5 3 1 2 3 4 5 1 3 5	1 3 5

2_4. Дан отсортированный массив различных целых чисел A[0..n-1] и массив целых чисел B[0..m-1]. Для каждого элемента массива B[i] найдите минимальный индекс элемента массива A[k], ближайшего по значению к B[i]. Время работы поиска для каждого элемента B[i]: O(log(k)).
n ≤ 110000, m ≤ 1000.

in	out
3 10 20 30 3 9 15 35	0 0 2
3 10 20 30 4 8 9 10 32	0 0 0 2

Задача № 3 (4 балла)

Во всех задачах из следующего списка следует написать структуру данных, обрабатывающую команды push* и pop*.

Формат входных данных.

В первой строке количество команд n. n ≤ 1000000.

Каждая команда задаётся как 2 целых числа: a b.

a = 1 - push front

a = 2 - pop front

a = 3 - push back

a = 4 - pop back

Команды добавления элемента 1 и 3 заданы с неотрицательным параметром b.

Для очереди используются команды 2 и 3. Для дека используются все четыре команды.

Если дана команда pop*, то число b - ожидаемое значение. Если команда pop вызвана для пустой структуры данных, то ожидается “-1”.

Формат выходных данных.

Требуется напечатать YES - если все ожидаемые значения совпали. Иначе, если хотя бы одно ожидание не оправдалось, то напечатать NO.

3_1. Реализовать очередь с динамическим зацикленным буфером.

in	out
3 3 44 3 50 2 44	YES
2 2 -1 3 10	YES
2 3 44 2 66	NO

3_2. Реализовать дек с динамическим зацикленным буфером.

in	out
3 1 44 3 50 2 44	YES
2 2 -1 1 10	YES
2 3 44 4 66	NO

3_3. Реализовать очередь с помощью двух стеков. Использовать стек, реализованный с помощью динамического буфера.

in	out
3 3 44 3 50 2 44	YES
2 2 -1 3 10	YES
2 3 44 2 66	NO

Задача № 4 (4 балла)

Решение всех задач данного раздела предполагает использование кучи, реализованной в виде класса.

4_1. Жадина.

Вовочка ест фрукты из бабушкиной корзины. В корзине лежат фрукты разной массы. Вовочка может поднять не более K грамм. Каждый фрукт весит не более K грамм. За раз он выбирает несколько самых тяжелых фруктов, которые может поднять одновременно, откусывает от каждого половину и кладет огрызки обратно в корзину. Если фрукт весит нечетное число грамм, он откусывает большую половину. Фрукт массы 1гр он съедает полностью.

Определить за сколько подходов Вовочка съест все фрукты в корзине.

Формат входных данных. Вначале вводится n - количество фруктов и n строк с массами фруктов. Затем K - "грузоподъемность".

Формат выходных данных. Неотрицательное число - количество подходов к корзине.

in	out
3 1 2 2 2	4
3 4 3 5 6	5
7 1 1 1 1 1 1 1 3	3

4_2. Быстрое сложение.

Для сложения чисел используется старый компьютер. Время, затрачиваемое на нахождение суммы двух чисел равно их сумме.

Таким образом для нахождения суммы чисел 1,2,3 может потребоваться разное время, в зависимости от порядка вычислений.

((1+2)+3) -> 1+2 + 3+3 = 9

((1+3)+2) -> 1+3 + 4+2 = 10

((2+3)+1) -> 2+3 + 5+1 = 11

Требуется написать программу, которая определяет минимальное время, достаточное для вычисления

суммы заданного набора чисел.
Формат входных данных. Вначале вводится n - количество чисел. Затем вводится n строк - значения чисел (значение каждого числа не превосходит 10^9, сумма всех чисел не превосходит 2*10^9).
Формат выходных данных. Натуральное число - минимальное время.

in	out
5 5 2 3 4 6	45
5 3 7 6 1 9	56

4_3. Тупики.
На вокзале есть некоторое количество тупиков, куда прибывают электрички. Этот вокзал является их конечной станцией. Дано расписание движения электричек, в котором для каждой электрички указано время ее прибытия, а также время отправления в следующий рейс. Электрички в расписании упорядочены по времени прибытия. Когда электричка прибывает, ее ставят в свободный тупик с минимальным номером. При этом если электричка из какого-то тупика отправилась в момент времени X, то электричку, которая прибывает в момент времени X, в этот тупик ставить нельзя, а электричку, прибывающую в момент X+1 — можно.
В данный момент на вокзале достаточное количество тупиков для работы по расписанию.
Напишите программу, которая по данному расписанию определяет, какое минимальное количество тупиков требуется для работы вокзала.
Формат входных данных. Вначале вводится n - количество электричек в расписании. Затем вводится n строк для каждой электрички, в строке - время прибытия и время отправления. Время - натуральное число от 0 до 10^9. Строки в расписании упорядочены по времени прибытия.
Формат выходных данных. Натуральное число - минимальное количеством тупиков.
Максимальное время: 50мс, память: 5Мб.

in	out
1 10 20	1
2 10 20 20 25	2
3 10 20 20 25 21 30	2

4_4. Скользящий максимум.
Дан массив натуральных чисел A[0..n), n не превосходит 10^8. Так же задан размер некоторого окна (последовательно расположенных элементов массива) в этом массиве k, k<=n. Требуется для каждого положения окна (от 0 и до n-k) вывести значение максимума в окне.
Скорость работы O(n log n), память O(n).
Формат входных данных. Вначале вводится n - количество элементов массива. Затем вводится n строк со значением каждого элемента. Затем вводится k - размер окна.
Формат выходных данных. Разделенные пробелом значения максимумов для каждого положения окна.

in	out

3 1 2 3 2	2 3
9 0 7 3 8 4 5 10 4 6 4	8 8 8 10 10 10

Задача № 5 (4 балла)

Во всех задачах данного раздела необходимо реализовать и использовать **сортировку слиянием**. Общее время работы алгоритма O(n log n).

5_1. Реклама.

В супермаркете решили оптимизировать показ рекламы. Известно расписание прихода и ухода покупателей (два целых числа). Каждому покупателю необходимо показать минимум 2 рекламы. Рекламу можно транслировать только в целочисленные моменты времени. Покупатель может видеть рекламу от момента прихода до момента ухода из магазина.

В каждый момент времени может показываться только одна реклама. Считается, что реклама показывается мгновенно. Если реклама показывается в момент ухода или прихода, то считается, что посетитель успел её посмотреть. Требуется определить минимальное число показов рекламы.

In	Out
5 1 10 10 12 1 10 1 10 23 24	5

5_2. Современники.

Группа людей называется современниками если был такой момент, когда они могли собраться вместе. Для этого в этот момент каждому из них должно было уже исполниться 18 лет, но ещё не исполниться 80 лет.

Дан список Жизни Великих Людей. Необходимо получить максимальное количество современников. В день 18летия человек уже может принимать участие в собраниях, а в день 80летия и в день смерти уже не может.

Замечание. Человек мог не дожить до 18-летия, либо умереть в день 18-летия. В этих случаях принимать участие в собраниях он не мог.

In	Out
3 2 5 1980 13 11 2055 1 1 1982 1 1 2030 2 1 1920 2 1 2000	3

5_3. Закраска прямой 1.

На числовой прямой окрасили *N* отрезков. Известны координаты левого и правого концов каждого отрезка (L_i и R_i). Найти длину окрашенной части числовой прямой.

In	Out
3 1 4 7 8 2 5	5

5_4. Закраска прямой 2.

На числовой прямой окрасили *N* отрезков. Известны координаты левого и правого концов каждого отрезка (L_i и R_i). Найти сумму длин частей числовой прямой, окрашенных ровно в один слой.

In	Out
3 1 4 7 8 2 5	3

Задача № 6 (3 балла)

Даны неотрицательные целые числа n,k и массив целых чисел из [0..10^9] размера n. Требуется найти k-ю порядковую статистику. т.е. напечатать число, которое бы стояло на позиции с индексом k (0..n-1) в отсортированном массиве. Напишите нерекурсивный алгоритм.

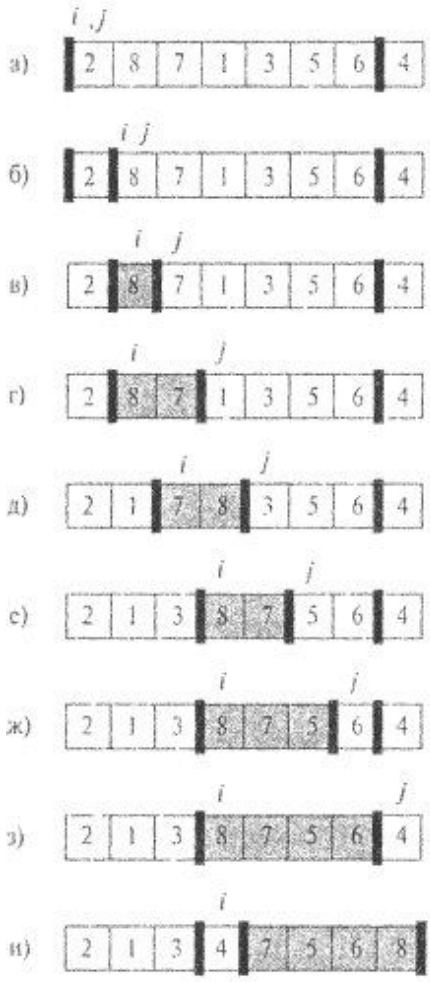
Требования к дополнительной памяти: O(n). Требуемое среднее время работы: O(n).

Функцию Partition следует реализовывать методом прохода двумя итераторами в одном направлении.

Описание для случая прохода от начала массива к концу:

- Выбирается опорный элемент. Опорный элемент меняется с последним элементом массива.
- Во время работы Partition в начале массива содержатся элементы, не бОльшие опорного. Затем располагаются элементы, строго бОльшие опорного. В конце массива лежат нерассмотренные элементы. Последним элементом лежит опорный.
- Итератор (индекс) i указывает на начало группы элементов, строго бОльших опорного.
- Итератор j больше i, итератор j указывает на первый нерассмотренный элемент.
- Шаг алгоритма. Рассматривается элемент, на который указывает j. Если он больше опорного, то сдвигаем j.

  Если он не больше опорного, то меняем a[i] и a[j] местами, сдвигаем i и сдвигаем j.
- В конце работы алгоритма меняем опорный и элемент, на который указывает итератор i.



6_1. Реализуйте стратегию выбора опорного элемента “медиана трёх”. Функцию Partition реализуйте методом прохода двумя итераторами от начала массива к концу.

6_2. Реализуйте стратегию выбора опорного элемента “медиана трёх”. Функцию Partition реализуйте методом прохода двумя итераторами от конца массива к началу.

6_3. Реализуйте стратегию выбора опорного элемента “случайный элемент”. Функцию Partition реализуйте методом прохода двумя итераторами от начала массива к концу.

6_4. Реализуйте стратегию выбора опорного элемента “случайный элемент”. Функцию Partition реализуйте методом прохода двумя итераторами от конца массива к началу.

In	Out
10 4 1 2 3 4 5 6 7 8 9 10	5

10 0 3 6 5 7 2 9 8 10 4 1	1
10 9 0 0 0 0 0 0 0 0 0 1	1

Задача № 7 (3 балла)

7_1. MSD для строк.

Дан массив строк. Количество строк не больше 10^5. Отсортировать массив методом поразрядной сортировки MSD по символам. Размер алфавита - 256 символов. Последний символ строки = '\0'.

In	Out
ab a aaa aa	a aa aaa ab

7_2. LSD для long long.

Дан массив неотрицательных целых 64-битных чисел. Количество чисел не больше 10^6. Отсортировать массив методом поразрядной сортировки LSD по байтам.

In	Out
3 4 1000000 7	4 7 1000000

7_3. Binary MSD для long long.

Дан массив неотрицательных целых 64-разрядных чисел. Количество чисел не больше 10^6. Отсортировать массив методом MSD по битам (бинарный QuickSort).

In	Out
3 4 1000000 7	4 7 1000000