Домашнее задание 2

Сдать задание нужно до 9 ноября (16:00).

Для заочников - тоже до 9 ноября (16:00).

Контест: <https://contest.yandex.ru/contest/14768/enter>

От каждой задачи нужно решить только один свой вариант. Варианты прописаны в ведомости:

<https://drive.google.com/open?id=1RqX6nsJVEaEKUrAOnhR847Z9rco44GT8Z_fJbUu1RP0>

# Задача № 3 (5 балла)

**3\_1. Первые k элементов длинной последовательности.**

Дана очень длинная последовательность целых чисел длины n. Требуется вывести в отсортированном виде её первые k элементов. Последовательность может не помещаться в память. Время работы O(n \* log(k)). Память O(k). Использовать слияние.

|  |  |
| --- | --- |
| In | Out |
| 9 4  3 7 4 5 6 1 15 4 2 | 1 2 3 4 |

**3\_2. Сортировка почти упорядоченной последовательности.**

Дана последовательность целых чисел a1...an и натуральное число k, такое что для любых i, j: если j >= i + k, то a[i] <= a[j]. Требуется отсортировать последовательность. Последовательность может быть очень длинной. Время работы O(n \* log(k)). Память O(k). Использовать слияние.

|  |  |
| --- | --- |
| In | Out |
| 10 4  0 4 3 2 1 8 7 6 5 9 | 0 1 2 3 4 5 6 7 8 9 |

**3\_3. Количество инверсий.**

Дана последовательность целых чисел из диапазона (-10^9 .. 10^9). Длина последовательности не больше 10^6. Числа записаны по одному в строке. Количество чисел не указано.

Пусть количество элементов n, и числа записаны в массиве a = a[i]: i из [0..n-1].

Требуется напечатать количество таких пар индексов (i,j) из [0..n-1], что (i < j и a[i] > a[j]).

Указание: количество инверсий может быть больше 4\*10^9 - используйте int64\_t.

#include <stdint.h>

int64\_t cnt = 0;

printf(“%ld”, cnt);

|  |  |
| --- | --- |
| In | Out |
| 1  2  3  4 | 0 |
| 4  3  2  1 | 6 |
| 3  2  2 | 2 |

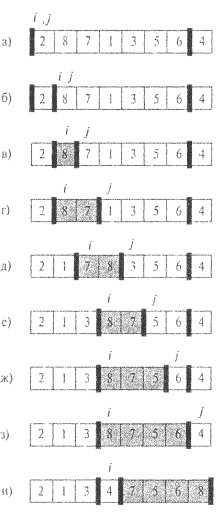
# Задача № 4 (5 баллов)

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.



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

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

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

**4\_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 |