Контрольная работа-2. Массивы и сортировка. 8Д.

В каждой из предложенных ниже задач, нужно реализовать те функции, которые описаны в условии. Функции должны называться, иметь точно такие же аргументы и возвращать те же значения, как написано в условии. В противном случае, система проверки не зачтет ваше решение.
Прототипы функций, которые нужно реализовать, писать **НЕ** надо. Система сама вставит их.
Также **НЕ** надо писать `main()`.
Если для реализации нужных функций вам понадобятся вспомогательные, используйте их. Для них заголовки писать можно и нужно.
К каждому решению система автоматически подключит библиотеки `stdio.h` и `iostream`. Если вам понадобятся другие библиотеки, подключайте их самостоятельно. Не стоит подключать `TXLib.h`, система вас не поймет.
Запрещается использовать стандартные функции сортировки, реализованные не вами.

A. *Третий максимум.* Реализуйте функцию
`int ThirdMax(int const array[], int size),`
которая находит третье по величине число массива. То есть число, которое будет стоять не третьем с конца месте, если массив отсортировать по возрастанию.
Гарантируется, что в массиве, к которому будет применяться функция, есть не менее трех элементов.
Изменять исходный массив (в частности, сортировать) запрещается.

B. *Полусумма соседей.* Реализуйте функцию
`void SumNeighborsTransform(double array[], int size),`
которая заменит каждый элемент на полусумму соседей (левого и правго). Нулевой и последний элементы остаются без изменений.
Также необходимо реализовать функцию
`void PrintArray(double const array[], int size),`
которая печатает массив `array` на одной строчке через пробел.

C. *Циклический сдвиг* Реализуйте функцию
`void RShift(int array[], int size),`
которая циклически передвинет все элементы массива на одну позицию вправо. То есть, нулевой элемент на первую позицию, первый элемент на вторую позицию, ... последний элемент на нулевую позицию.
Также необходимо реализовать функцию
`void PrintArray(int const array[], int size),`
которая печатает массив `array` на одной строчке через пробел.

D. *Количество уникальных элементов* Вам необходимо найти количество уникальных элементов массива. То есть тех, которые встречаются ровно один раз.
Один из способов решить эту задачу — отсортировать массив, далее в отсортированном массиве найти количество уникальных элементов.
Реализуйте две функции
`void MySort(int array[], int size),` и
`int SortedArrayUniqueCount(int const array[], int size),`

E. *Великий волшебник.* У Вани есть набор карточек, на которых написаны числа. Он научился показывать следующий фокус — перемешивать карты, затем выкладывать поочередно по одной карте сверху и снизу колоды, пока карты не кончатся. В результате чего последовательность чисел, написанных на выложенных картах, оказывается отсортированной по возрастанию.
Реализуйте функцию
`void StrangeSort(int array[], int size),`
которая сортирует элементы массива в таком странном порядке. Минимальный элемент должен стоять на нулевом месте, второй минимум на последнем, третий минимум на первом, и так далее.
Также необходимо реализовать функцию
`void PrintArray(int const array[], int size),`
которая печатает массив `array` на одной строчке через пробел.

F. *Поразрядная сортировка*
Поразрядная сортировка является одним из видов сортировки, которые работают за линейное от размера сортируемого массива время. Такая скорость достигается за счет того, что эта сортировка использует внутреннюю структуру сортируемых объектов. Изначально этот алгоритм использовался для сортировки перфокарт. Первая его компьютерная реализация была создана в университете MIT Гарольдом Сьюардом (Harold H. Seward). Опишем алгоритм подробнее.
Пусть задан массив строк $s_1, \ldots, s_i$ причем все строки имеют одинаковую длину m. Работа алгоритма состоит из m фаз. На i-ой фазе строки сортируются по i-ой с конца букве. Происходит это следующим образом. В этой задаче рассматриваются строки из цифр от 0 до 9. Для каждой цифры создается "корзина" ("bucket"), после чего строки s_i распределяются по "корзинам" в соответствии с i-ой с конца цифрой. Строки, у которых i-ая с конца цифра равна j попадают в j-ую корзину (например, строка 123 на первой фазе попадет в третью корзину, на второй — во вторую, на третьей — в первую). После этого элементы извлекаются из корзин в порядке увеличения номера корзины.
Таким образом, после первой фазы строки отсортированы по последней цифре, после двух фаз — по двум последним,..., после m фаз — по всем. При этом важно, чтобы элементы в корзинах сохраняли тот же порядок, что и в исходном массиве (до начала этой фазы). Например, если массив до первой фазы имеет вид: $111, 112, 211, 311$, то элементы по корзинам распределятся следующим образом: в первой корзине будет $111, 211, 311$, а второй: 112.

G. *K-ая порядковая статистика.* Напишите функцию, которая находит k-ое по величине число массива. Функция не должна менять исходный массив и работать за линейное время.