Лабораторная работа №1

По курсу «Языки программирования и методы программирования» (информатика, 3 семестр)

Цель: написать алгоритмы сортировки и проанализировать их работу на различных данных.

Реализованные следующие **алгоритмы сортировок**:

1. BubbleSort (пузырьковая сортировка)
2. ShakerSort(модификация метода пузырька)
3. MergeSort (сортировка слиянием)
4. HeapSort (пирамидальная сортировка)
5. QuickSort (быстрая сортировка с опорным случайным элементом)

Возможна *генерация* следующих наборов данных:

1. Случайные значения
2. Массив возрастающих элементов
3. Массив убывающих элементов

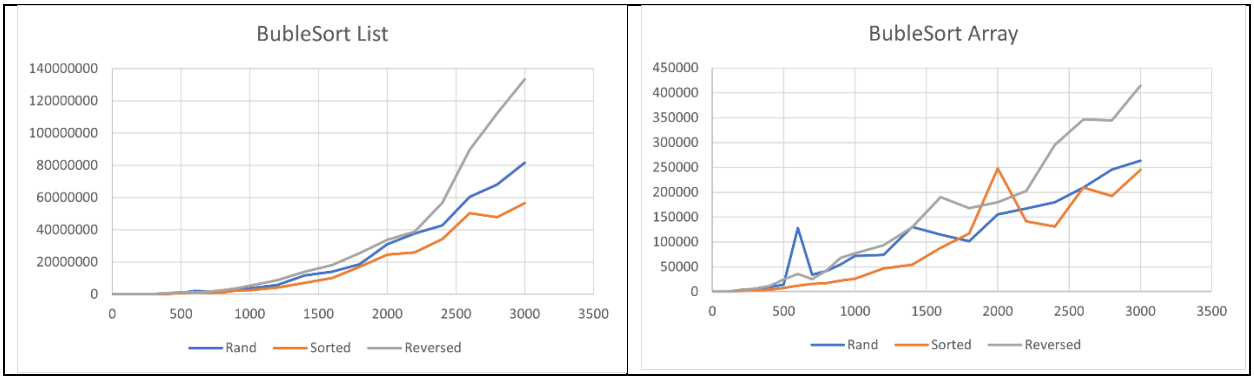
Тестирование проводилось на:

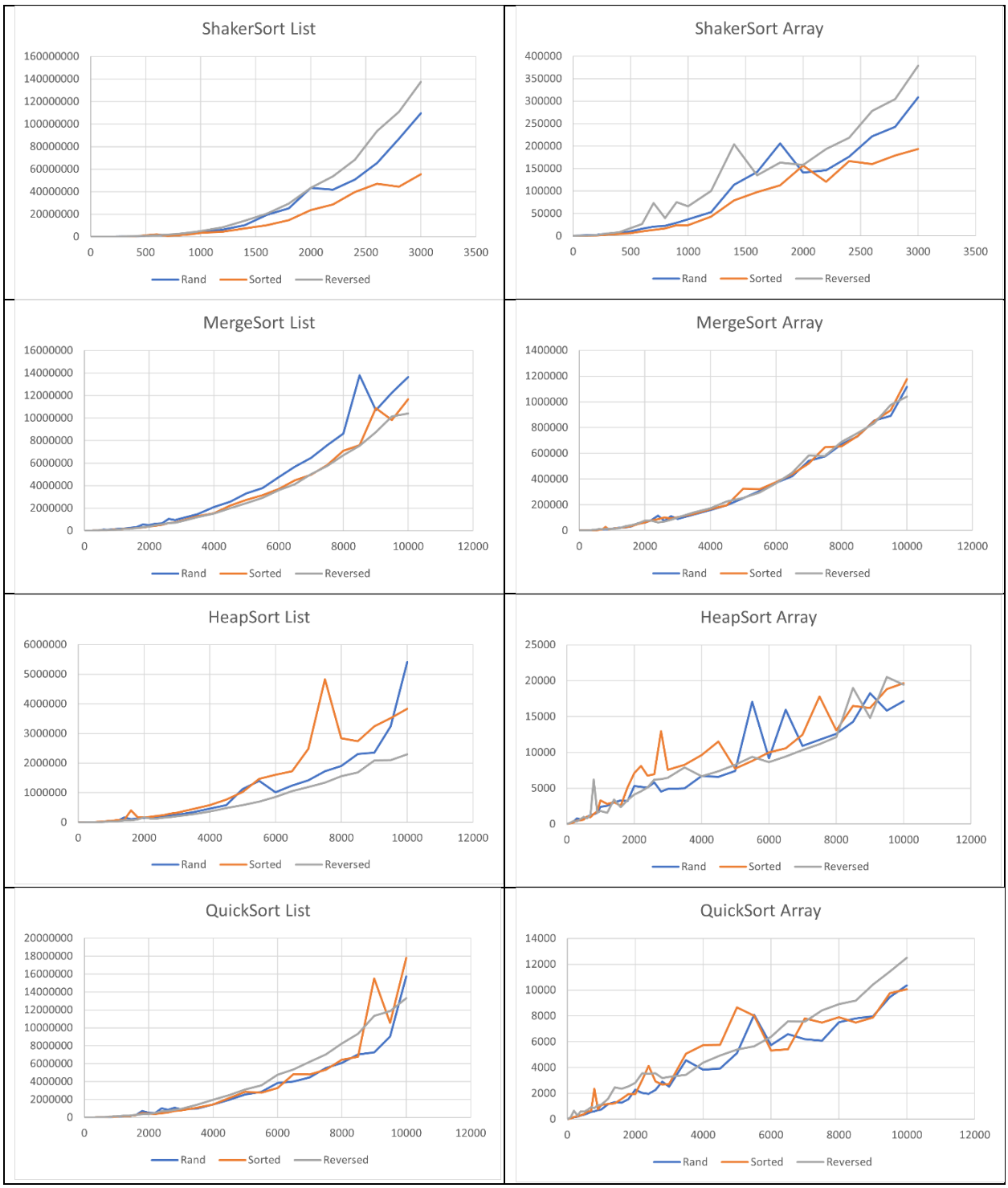
- (для медленных сортировок – квадратичных) числах от 0 до 3000, от 0 до 1000 шаг 100, от 1000 до 3000 шаг 200.
- (для быстрых сортировок – n log(n)) числах от 0 до 10000, от 0 до 1000 шаг 100, от 1000 до 3000 шаг 200, от 3000 до 1000 шаг 500.

Значения были выбраны исходя из возможностей компьютера. Далее по графикам можно будет увидеть, что где-то можно было бы повысить верхнюю границу, однако она выбиралась исходя из самого медленного алгоритма группы, чтобы в последствие было удобнее сравнивать графики разных сортировок.

Код тестов можно посмотреть в «graphtests.cpp», результаты измерения времени и графики в «tests.xlsx». Графики были построены с помощью Excel.

Графики зависимости времени от количества элементов:





Как видно по графикам, алгоритмы на ListSequence работают сильно медленнее, чем ArraySequence (до 100 раз, например, на быстрой сортировке). Это можно объяснить тем, что чаще всего вызывается функция обращения к элементу по индексу, что в ListSequence происходит за O(n), а в ArraySequence за O(1).

Квадратичные сортировки показали себя практически одинаково на случайных данных, однако на отсортированных в обратном порядке данных ShakerSort работает медленнее.

QuickSort лучше всего работает среди быстрых сортировок.