Задания на практическую работу 03. Алгоритмы сортировки. Часть 2.

Провести практическое исследование эффективности алгоритмов сортировки: выбором, вставками, пузырьком, слиянием, Шелла (с разными последовательностями, см. свой вариант)

Каждый алгоритм сортировки реализовать в виде отдельной функции(й), все реализации разместить в одном отдельном модуле (.h + .cpp файл, **модули (`module`) стандарта C++20 НЕ использовать**)

В другом модуле (.h + .cpp файлы) разместить вспомогательные функции для работы с массивами: вывод на экран и в файл, генерация тестовых данных, чтение данных, проверки отсортированности, …

Основная программа должна

1. Сгенерировать и сохранить на диск файлы с наборами целых чисел.
   Наборы генерировать согласно Вашему варианту, в каждом варианте по 12 файлов **(см. в конце задания).**
   Наборы должны содержать разное количество чисел N, с разными распределениями **(повторяющиеся числа в наборе разрешены)**.
   a) равномерно распределенные числа из диапазона [a, b]
   - a100.txt, количество N = 100
   - a1000.txt, количество N = 1000
   - a10000.txt, количество N = 10000
   - a50000.txt, количество N = 50000
   - a100000.txt, количество N = 100000

   b) отсортированные по возрастанию числа из диапазона [a, b]
   - b100.txt, количество N = 100
   - b1000.txt, количество N = 1000
   - b10000.txt, количество N = 10000
   - b50000.txt, количество N = 50000
   - b100000.txt, количество N = 100000

   c) отсортированные по убыванию числа из диапазона [a, b]
   - c100.txt, количество N = 100
   - c1000.txt, количество N = 1000
   - c10000.txt, количество N = 10000
   - c50000.txt, количество N = 50000
   - c100000.txt, количество N = 100000

   d*) частично отсортированные числа из диапазона [a, b] (95% чисел в файле упорядочены по возрастанию, но 5% чисел размещены с нарушением порядка; для этого сгенерируйте полностью отсортированный массив, а затем выполните `N * 0.05` случайных обменов (`swap`) между элементами)
   - d100.txt, количество N = 100
   - d1000.txt, количество N = 1000
   - d10000.txt, количество N = 10000
   - d50000.txt, количество N = 50000
   - d100000.txt, количество N = 100000

   e*) отсортированные блоки чисел из диапазона [a, b] (весь набор данных должен быть разбит на N/M блоков; каждый блок длиной M элементов содержит отсортированные числа; блоки между собой НЕ обязаны быть отсортированы; числа в разных блоках могут повторяться)
   - e100.txt, количество N = 100, M = 20
   - e1000.txt, количество N = 1000, M = 50
   - e10000.txt, количество N = 10000, M = 1000
   - e50000.txt, количество N = 50000, M = 2000
   - e100000.txt, количество N = 100000, M = 5000

2. Из сгенерированных файлов прочитать данные в динамические массивы C++ **(std::vector не использовать)**
3. Провести серию из 10 запусков каждого алгоритма сортировки на каждом наборе данных. При этом:

   - Запуск алгоритмов проводить на копиях исходных прочитанных данных, не сортировать повторно уже отсортированное!

   - После каждой сортировки проверять что массив действительно отсортирован вызовом созданной для этой проверки функции. Если сортировка не получилась, то результат этого запуска не учитывается в дальнейшей статистике, а сам запуск надо повторить.

   - Сортировки поиском, вставками, пузырьком тестировать только для наборов с N <= 50000, сортировки Шелла и слиянием – запускать на всех наборах
   Замечание1: если на Вашем компьютере для тестового набора с N=50000 время выполнения сортировок превышает 300 секунд, разрешается использовать набор с N=10000 вместо набора с N=50000.

   Для каждого запуска фиксировать
   - **время выполнения сортировки в микросекундах** (не должно включать время, потраченное на создание копии исходных данных, а также время, потраченное на операции вывода на экран или в файл и прочие, не относящиеся непосредственно к сортировке операции)
   - **количество выполненных операций сравнения** (каждое логическое сравнение двух элементов массива (arr[i] < arr[j])) инкрементирует счётчик; сравнения индексов, счетчиков, границ и т.п. не учитываются).
   - **количество выполненных операций обмена** (обменом считать каждый swap или перемещение элемента внутри массива на другую позицию, для сортировки слиянием обменами считать операции копирования из вспомогательного массива)

   Замечание2: число сравнений/обменов может быть порядка $O(N^2)$, для N=100000 это может превысить 32-бит, поэтому для счетчиков рекомендуется использовать uint64_t или unsigned long long

   Для каждого размера датасета N, каждого набора данных и каждого алгоритма сортировки вывести на экран и в файл **results.txt**
   - среднее время выполнения (по 10 тестовым запускам)
   - среднее количество сравнений
   - среднее количество обменов

Примерный формат вывода:

```
N = 1000
Dataset: равномерное распределение
                    Bubble      Insertion      Selection          Shell          Merge
-------------------------------------------------------------------------------------
       Время          2886            662            990             92            158
  Сравнений        499455         242833         499500          15794           8687
    Обменов        241838         241838            995           8305           9976
-------------------------------------------------------------------------------------
Dataset: отсортировано по возрастанию
                    Bubble      Insertion      Selection          Shell          Merge
-------------------------------------------------------------------------------------
       Время             4              6           1090             25            105
  Сравнений           999            999         499500           8006           5044
    Обменов             0              0              0              0           9976
-------------------------------------------------------------------------------------
Dataset: отсортировано по убыванию
                                       ...
-------------------------------------------------------------------------------------

N = 50000
Dataset: равномерное распределение
                    Bubble      Insertion      Selection          Shell          Merge
-------------------------------------------------------------------------------------
       Время       8022763        1707887        2411415           9598           9409
```

```
   Сравнений     1249960122      626270858     1249975000        1875019         718096
     Обменов      626220872      626220872          49986        1200502         784464
-------------------------------------------------------------------------------------------
Dataset: отсортировано по возрастанию
                     Bubble      Insertion      Selection          Shell          Merge
-------------------------------------------------------------------------------------------
       Время            213            271        2401059           2250           6072
   Сравнений          49999          49999     1249975000         700006         401952
     Обменов              0              0              0              0         784464
-------------------------------------------------------------------------------------------
                                        ...
N = 100000
Dataset: равномерное распределение
                      Shell          Merge
-----------------------------------------
       Время          20811          18624
   Сравнений        4054288        1536203
     Обменов        2604824        1668928
-------------------------------------------------------------------------------------------
Dataset: отсортировано по возрастанию
                      Shell          Merge
-----------------------------------------
       Время           4860          12685
   Сравнений        1500006         853904
     Обменов              0        1668928
-------------------------------------------------------------------------------------------
...
```

При защите работы быть готовым показать и объяснить

- принципы работы сортировок Шелла и слиянием
- вывод на экран и в файл отсортированных массивов (или их отдельных частей для больших N) на любой стадии работы каждого алгоритма сортировки
- интерпретацию полученных результатов: почему для разных алгоритмов и наборов данных получаются именно такие замеры времени и количества операций
- каждый фрагмент программного кода Вашего решения

Номер Вашего варианта такой же, как в задании на практическую работу 02. Алгоритмы сортировки. Часть 1.

Вариант 1.

Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла с простой стратегией убывания gap = n/2, n/4, … 1
Тестовые последовательности: числа из диапазонов
[0, 1000) для N=1000; [0, 50000) для N=50000; [0, 100000) для N=100000;
сохранять в файлах
a1000.txt, a50000.txt, a100000.txt
b1000.txt, b50000.txt, b100000.txt
c1000.txt, c50000.txt, c100000.txt
e1000.txt, e50000.txt, e100000.txt

Вариант 2.

Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Циура (Чиуры, Ciura)
Тестовые последовательности: числа из диапазонов
[900, 1000) для N=100; [900, 10900) для N=10000; [900, 100900) для N=100000;
сохранять в файлах
a100.txt, a10000.txt, a100000.txt
b100.txt, b10000.txt, b100000.txt
c100.txt, c10000.txt, c100000.txt
d100.txt, d10000.txt, d100000.txt

Вариант 3.
Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Токуды (Tokuda)
Тестовые последовательности: числа из диапазонов
[0, 1000) для N=1000; [0, 50000) для N=50000; [0, 100000) для N=100000;
сохранять в файлах
a1000.txt, a50000.txt, a100000.txt
b1000.txt, b50000.txt, b100000.txt
c1000.txt, c50000.txt, c100000.txt
d1000.txt, d50000.txt, d100000.txt

Вариант 4.
Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Кнута (Knuth)
Тестовые последовательности: числа из диапазонов
[0, 1000) для N=1000; [0, 50000) для N=50000; [0, 100000) для N=100000;
сохранять в файлах
a1000.txt, a50000.txt, a100000.txt
b1000.txt, b50000.txt, b100000.txt
c1000.txt, c50000.txt, c100000.txt
d1000.txt, d50000.txt, d100000.txt

Вариант 5.
Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Пратта (Pratt)
Тестовые последовательности: числа из диапазонов
[0, 100) для N=100; [0, 10000) для N=10000; [0, 100000) для N=100000;
сохранять в файлах
a100.txt, a10000.txt, a100000.txt
b100.txt, b10000.txt, b100000.txt
c100.txt, c10000.txt, c100000.txt
e100.txt, e10000.txt, e100000.txt

Вариант 6.
Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Хиббарда (Hibbard)
Тестовые последовательности: числа из диапазонов
[0, 100) для N=100; [0, 50000) для N=50000; [0, 100000) для N=100000;
сохранять в файлах
a100.txt, a50000.txt, a100000.txt
b100.txt, b50000.txt, b100000.txt
c100.txt, c50000.txt, c100000.txt
d100.txt, d50000.txt, d100000.txt

Вариант 7.
Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Ли (Lee)
Тестовые последовательности: числа из диапазонов
[0, 1000) для N=1000; [0, 50000) для N=50000; [0, 100000) для N=100000;
сохранять в файлах
a1000.txt, a50000.txt, a100000.txt
b1000.txt, b50000.txt, b100000.txt
c1000.txt, c50000.txt, c100000.txt
e1000.txt, e50000.txt, e100000.txt

Вариант 8.
Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Седжвика (Sedgewick)
Тестовые последовательности: числа из диапазонов
[0, 1000) для N=1000; [0, 50000) для N=50000; [0, 100000) для N=100000;
сохранять в файлах
a1000.txt, a50000.txt, a100000.txt
b1000.txt, b50000.txt, b100000.txt
c1000.txt, c50000.txt, c100000.txt
d1000.txt, d50000.txt, d100000.txt

Вариант 9.
Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Кнута (Knuth)
Тестовые последовательности: числа из диапазонов
[100, 200) для N=100; [100, 50100) для N=50000; [100, 100100) для N=100000;
сохранять в файлах
a100.txt, a50000.txt, a100000.txt
b100.txt, b50000.txt, b100000.txt
c100.txt, c50000.txt, c100000.txt
e100.txt, e50000.txt, e100000.txt

Вариант 10.
Сортировка выбором, вставками, пузырьком, слиянием,
сортировка Шелла на основе последовательности Гоннета и Бэя (Gonnet & Baeza-Yates)
Тестовые последовательности: числа из диапазонов
[0, 1000) для N=1000; [0, 50000) для N=50000; [0, 100000) для N=100000;
сохранять в файлах
a1000.txt, a10000.txt, a100000.txt
b1000.txt, b10000.txt, b100000.txt
c1000.txt, c10000.txt, c100000.txt
e1000.txt, e10000.txt, e100000.txt