МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних наук та інформаційних технологій
Кафедра програмного забезпечення

ЗВІТ

до лабораторної роботи №1
на тему: «Метод сортування бульбашкою»
з дисципліни: «Алгоритми і структури даних»

Лектор:
доц. кафедри ПЗ
Коротєєва Т. О.

Виконав:
ст. гр. ПЗ-22
Чаус О. М.

Прийняв:
асис. кафедри ПЗ
Франко А. В.

« ____ » _____ 2022 р.

∑= _____ .....:

Львів-2022

Тема роботи: Метод сортування бульбашкою.

Мета роботи: Вивчити алгоритм сортування бульбашкою. Здійснити програмну реалізацію алгоритму сортування бульбашкою. Дослідити швидкодію алгоритму сортування бульбашкою.

Теоретичні відомості

Сортування бульбашкою (англійською «Bubble Sort») — простий алгоритм сортування. Алгоритм працює наступним чином — у поданому наборі даних (списку чимасиві) порівнюються два сусідні елементи. Якщо один з елементів, не відповідає критерію сортування (є більшим, або ж, навпаки, меншим за свого сусіда), то ці два елементи міняються місцями. Прохід по списку продовжується до тих пір, доки дані не будуть відсортованими. Алгоритм отримав свою назву від того, що процес сортування згідно нього нагадує поведінку бульбашок повітря у резервуарі з водою. Оскільки для роботи з елементами масиву він використовує лише порівняння, це сортування на основі порівнянь.

Покрокове зображення алгоритму

АЛГОРИТМ BU. Сортування масиву алгоритмом бульбашки з прапорцем в порядку зростання. **arr** – вхідний масив, **i**, **j** – індекси елементів масиву, **sorted** - прапорець.

BU1. Якщо розмір масиву = 0, перехід до

BU2. Ініціалізація **i** = 0, **sorted** = **false**.

BU3. Ініціалізація **j** = 0.

BU4. Якщо **arr[j]> arr[j+1]**, поміняти місцями **arr[j]** і **arr[j+1]**, **sorted = true**.

BU5. Інкрементація **j = j + 1**. Якщо **j** ≥ **arr - i**, перехід до **BU3**, інакше перехід до **BU6**.

BU6. Якщо **sorted = true**, перехід до **BU8**, інакше перехід до **BU7**.

BU7. Інкрементація **i = i + 1**. Якщо **i** ≥ розмір **arr**, перехід до **BU8**, інакше перехід до **BU8**. Вихід з алгоритму.

Індивідуальне завдання

Написати консольну програму на мові програмування С, дотримуючись стандатру ANSI C99. Реалізована програма повинна виконувати наступну послідовність дій:

1) запитуватиме в користувача кількість цілих чотирьохбайтових знакових чисел — елементів масиву, сортування якого буде пізніше здійснено;
2) виділятиме для масиву стільки пам’яті, скільки необхідно для зберігання вказаної кількості елементів, але не більше;
3) ініціалізовуватиме значення елементів масиву за допомогою стандартної послідовності псевдовипадкових чисел;
4) засікатиме час початку сортування масиву з максимально можливою точністю;
5) сортуватиме елементи масиву в неспадному порядку за допомогою алгоритму сортування бульбашкою;
6) засікатиме час закінчення сортування масиву з максимально можливою точністю;
7) здійснюватиме перевірку упорядкованості масиву;
8) повідомлятиме користувачу результат перевірки упорядкованості масиву та загальний час виконання сортування з максимально можливою точністю;
9) звільнятиме усю виділену раніше пам’ять.

Варіант 12:

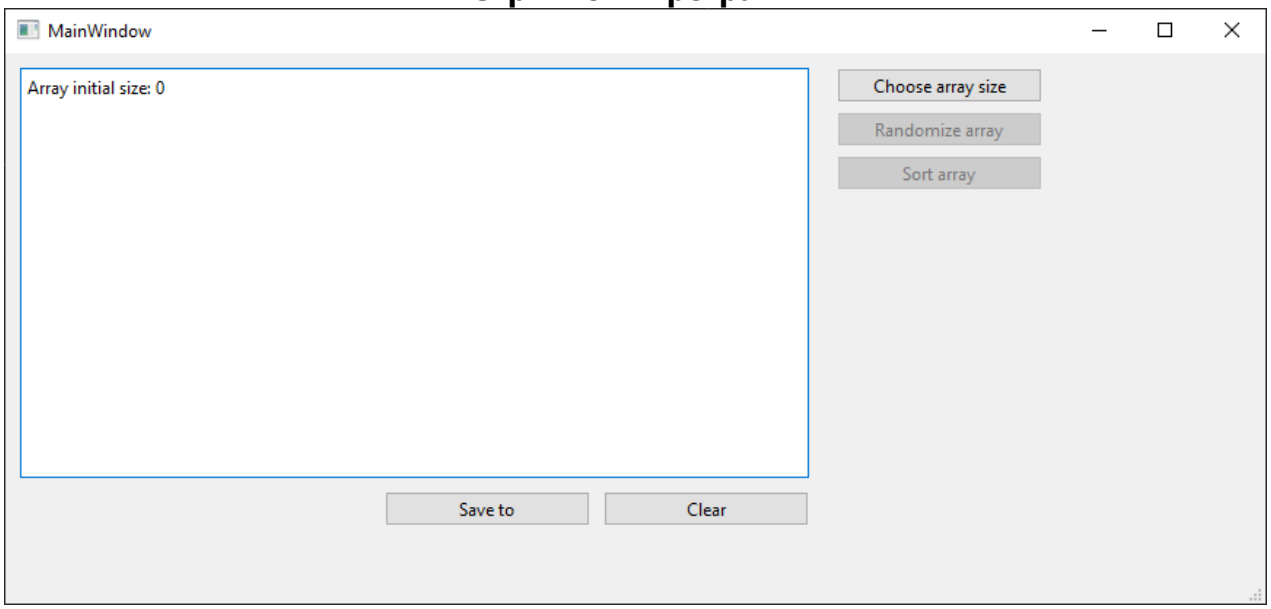
Задано одномірний масив дійсних чисел. Впорядкувати елементи, розташовані після максимального елемента в порядку спадання.

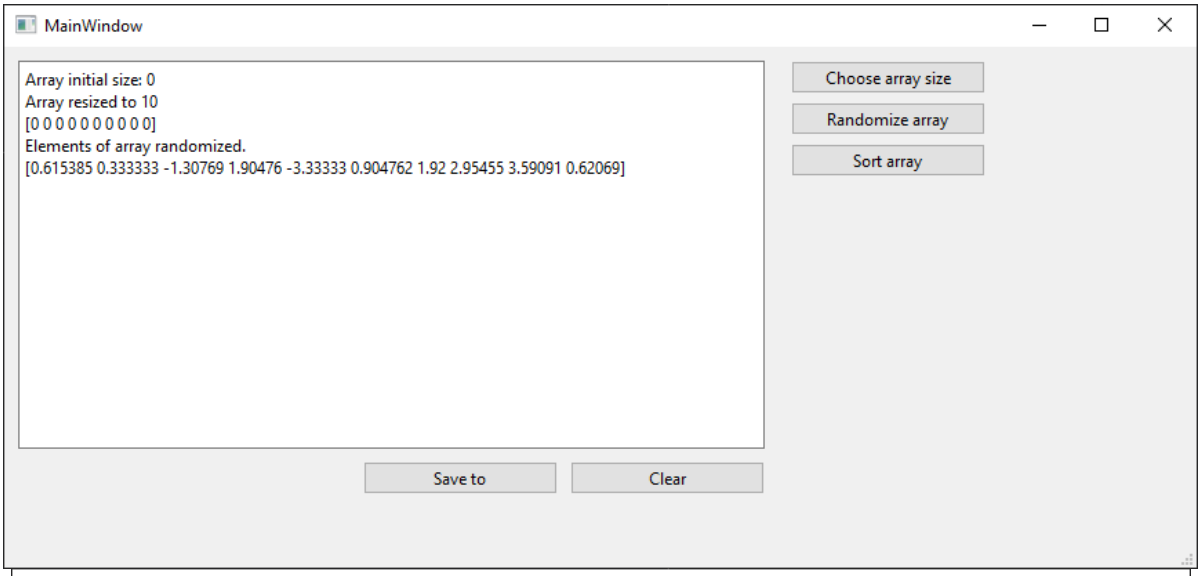
Код програми

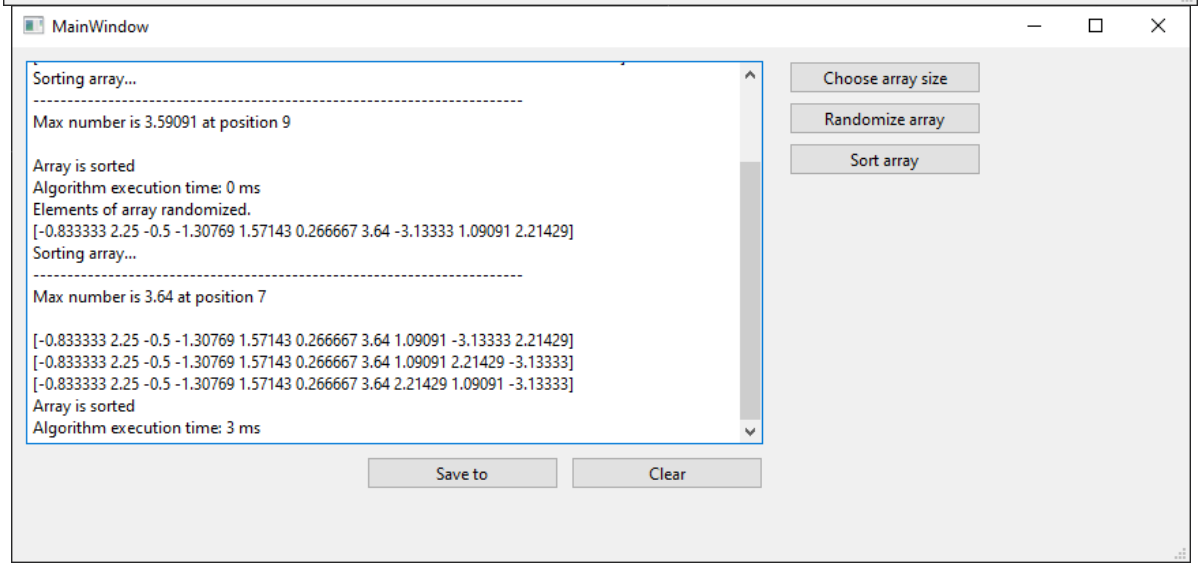
```
...
void MainWindow::on_b_sort_clicked()
{
    ui->textEdit->append("Sorting array...\n"
                         "-------------------------------------------------------
------------------");
    double max_value = main_vector[0];
    std::size_t max_index = 0;
    for(std::size_t i = 1; i < main_vector.size(); i++)
        if(main_vector[i] > max_value){
            max_value = main_vector[i];
            max_index = i;
        }
    ui->textEdit->append("Max number is " + QString::number(max_value) + " at
position " + QString::number(max_index + 1) + "\n");
    auto time1 = std::chrono::high_resolution_clock::now();
    for(std::size_t i = 0; i < main_vector.size() - max_index; i++)
    {
        bool swapped = false;
        for(std::size_t j = max_index; j < main_vector.size() - i - 1; j++){
            if(main_vector[j] < main_vector[j + 1]) {
                std::swap(main_vector[j], main_vector[j + 1]);
                swapped = true;
                PrintVector(main_vector);
            }
        }
        if (!swapped)
            break;
    }
    auto time2 = std::chrono::high_resolution_clock::now();
    ui->textEdit->append("Array is sorted\nAlgorithm execution time: " +

QString::number(std::chrono::duration_cast<std::chrono::milliseconds>(time2 -
time1).count()) + " ms");
}
...
```

Скріншоти програми







Висновок: під час виконання лабораторної роботи вивчив алгоритм сортування бульбашкою. Здійснив програмну реалізацію алгоритму сортування бульбашкою. Дослідив швидкодію алгоритму сортування бульбашкою, що дорівнює $O(n^2)$