Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Інститут комп’ютерних наук та інформаційних технологій

Кафедра програмного забезпечення



**Звіт**

Про виконання лабораторної роботи №1

**на тему:**

«Метод сортування бульбашкою»

З дисципліни «Алгоритми і структури даних»

**Лектор:**

доц. каф. ПЗ

Коротєєва Т.О.

**Виконав:**

ст. гр. ПЗ-26

Комарницький В.В.

**Прийняв:**

асистент каф. ПЗ

Симець І.І.

« \_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑ = \_\_\_\_\_ \_\_\_\_\_\_\_\_\_ .

Львів-2022

**Тема роботи:** Метод сортування бульбашкою.

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

**Теоретичний матеріал**

Алгоритм сортування бульбашкою ґрунтується на ідеї спливаючої бульбашки. В цьому алгоритмі елементи міняються місцями, як тільки виявлено, що порушено порядок між ними.

У найбільш невдалому випадку складність алгоритму дорівнює О(n²) (n – кількість елементів у вибірці для сортування), однак при великих n цей алгоритм має низьку ефективність.

Ідея алгоритму полягає в тому, що при виконанні проходження методом бульбашки не було здійснено жодного обміну елементів масиву, то це означає, що масив вже посортований і інші кроки проходження по масиві не потрібні.

**Покроковий опис роботи алгоритму сортування бульбашкою з використанням прапорця:**

**Алгоритм BS** сортує елементи одновимірного масиву в порядку спадання.

Задано масив елементів

**BS1.**  Цикл за індексом проходження. Повторювати кроки проходження B2 i B3 при і=1…n-1.

**BS2.** Встановлення значення для прапорця перестановки – flag = 0.

**BS3.** Виконання проходження. Якщо ,то присвоюємо змінній flag = 1і міняємо місцями елементи масиву ; якщо після проходження flag = 0, то завершення виконання алгоритму.

**BS4.** Кінець. Вихід.

**Індивідуальне завдання**

Варіант 12.

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

**Код програми**

**Функція, яка реалізує алгоритм сортування бульбашкою.**

­­­­-­­­­

void MainWindow::**bubbleSort**(double \*arr) {

QString steps;

int flag;

*for*(int i = 0; i < size1; ++i){

flag = 0;

*for*(int j = 0; j < size1 - i - 1; ++j)

{

*if*(arr[j] < arr[j+1])

{

double tmp = arr[j+1];

arr[j+1] = arr[j];

arr[j] = tmp;

flag = 1;

}

}

steps += "\n\n Step" + QString::number(i) + "\n";

*for*(int k = 0; k < size1; ++k)

{

steps += (QString::number(arr[k]) + ", ");

}

steps += "\n";

*if*(flag == 0)

{

*break*;

}

}

ui->steps\_list->setText(steps);

}­­­

**Функція введення масиву з клавіатури**

void MainWindow::**outputListArray**() {

int sizeArray = 1;

QStringList list = ui->line\_enter->text().split(QLatin1Char(';'), Qt::*SkipEmptyParts*);

sizeArray = list.length();

double ourArray[sizeArray];

QString givenArrayString = "Our array = [ ";

double max = list[0].toDouble();

int index = 0;

*for* (int i = 0; i < sizeArray; ++i)

{

ourArray[i] = list[i].toDouble();

*if*(max < ourArray[i]){

max = ourArray[i];

index = i;

}

}

*delete* [] array;

size1 = sizeArray-index;

array = *new* double[size1];

int k = -1;

*for*(int j = index; j < sizeArray; ++j)

{

k += 1;

*for*(int i = k; i < size1+1; ++i)

{

array[i] = ourArray[j];

*if*(i == 0)

{

givenArrayString += QString::number(array[i]);

*break*;

}

givenArrayString += (", " + QString::number(array[i]));

*break*;

}

}

givenArrayString += " ]";

ui->list\_array->setText(givenArrayString);

}

**Функція для заповнення масиву випадковими числами.**

void MainWindow::**outputRandomList**()

{

int numberElements = ui->spinBox\_number->value();

double ourArray[numberElements];

QString givenArrayString = "Our array = [ ";

*delete* [] array;

std::uniform\_real\_distribution<double> distribution (-50000., 50000.);

QString initialArray = "[ ";

*for*(int i = 0; i < numberElements; ++i)

{

ourArray[i] = distribution(\*QRandomGenerator::global());

*if*(i == 0)

{

initialArray += QString::number(ourArray[i]);

}

initialArray += (", " + QString::number(ourArray[i]));

}

initialArray += " ]";

ui->steps\_list->setText(initialArray);

double max = ourArray[0];

int index = 0;

*for*(int i = 0; i < numberElements; ++i)

{

*if*(max < ourArray[i])

{

max = ourArray[i];

index = i;

}

}

size1 = numberElements-index;

array = *new* double[size1];

int k = -1;

*for*(int j = index; j < numberElements; ++j)

{

k += 1;

*for*(int i = k; i < size1+1; ++i)

{

array[i] = ourArray[j];

*if*(i == 0)

{

givenArrayString += QString::number(array[i]);

*break*;

}

givenArrayString += (", " + QString::number(array[i]));

*break*;

}

}

givenArrayString += " ]";

ui->list\_array->setText(givenArrayString);

}

**Протокол виконання програми**

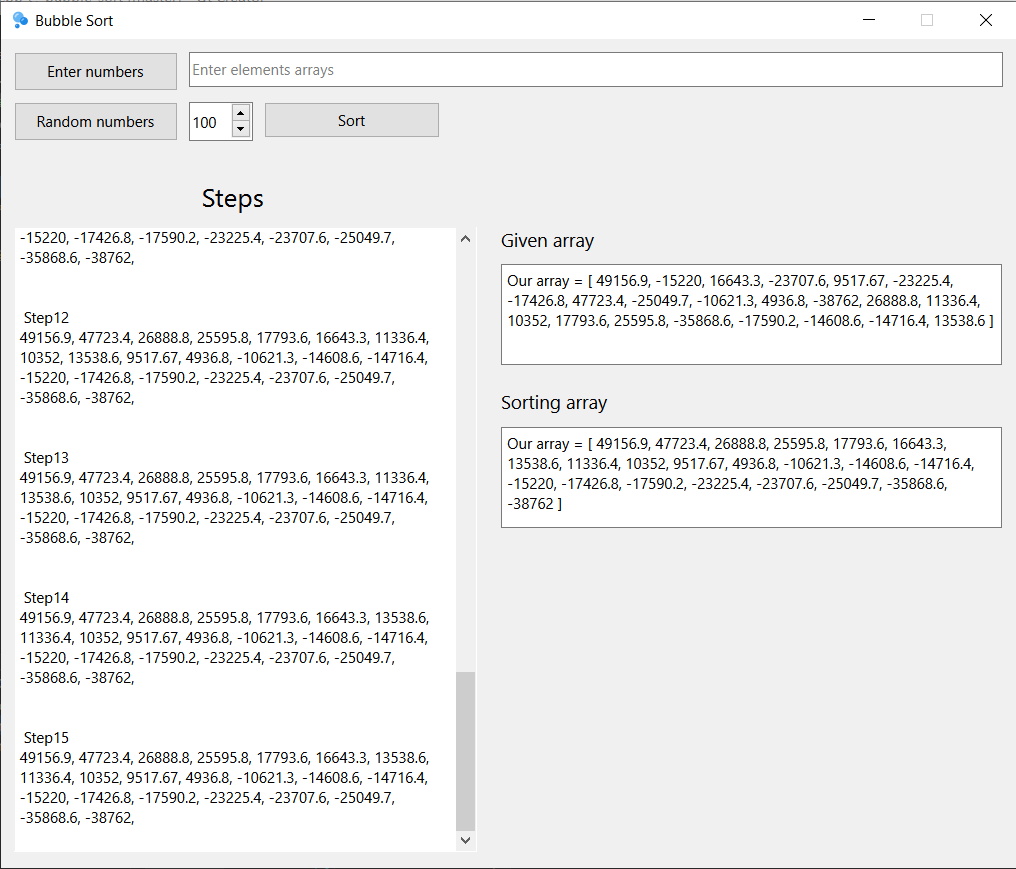


Рис.1. Сортування масиву розміром 100, який

заповнений випадковими числами.

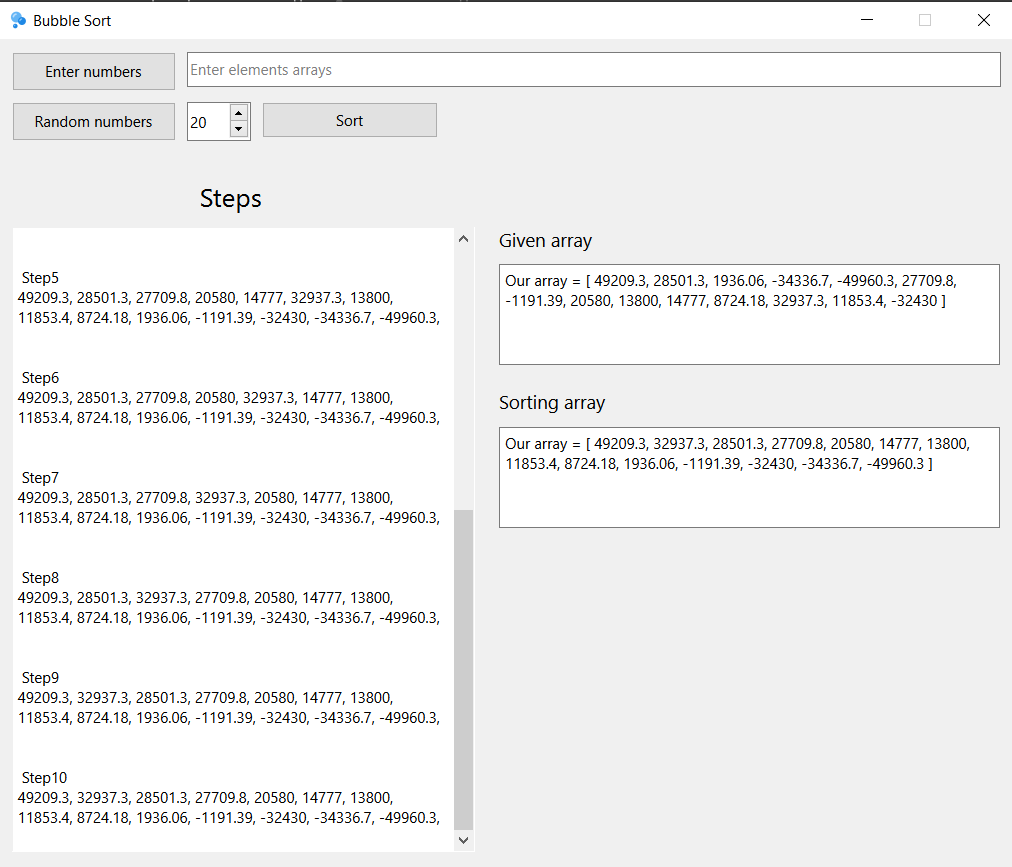


Рис.2. Сортування масиву розміром 20, який

заповнений випадковими числами.

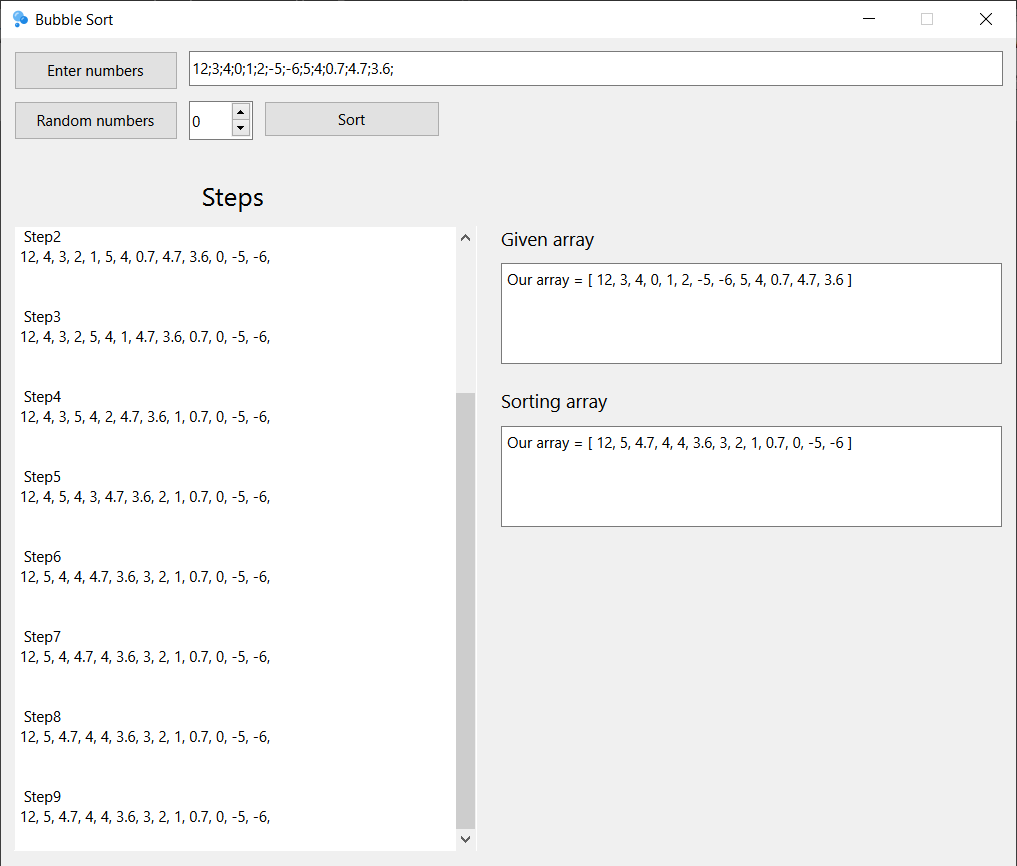


Рис.3. Сортування масиву, який

заповнений числами введеними з клавіатури.

**Висновки**

Під час виконання лабораторної роботи №1 я ознайомився з алгоритмом сортування бульбашкою, реалізував його на практиці та навчився описувати алгоритми.