**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

Інститут **ІКНІ**

Кафедра **ПЗ**



**ЗВІТ**

**До лабораторної роботи №6**

**На тему:** *“Метод сортування підрахунком”*

**З дисципліни:** *“Алгоритми і структури даних”*

**Лектор:**

Доцент кафедри ПЗ

Коротєєва Т.О.

**Виконав:**

ст. гр. ПЗ-21

Мартинюк Н.В.

**Прийняв:**

Асистент кафедри ПЗ

Симець І.І.

«\_\_» \_\_\_\_\_\_\_\_\_ 2020 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів-2020

**Тема:** Метод сортування підрахунком.

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

**Завдання для лабораторної роботи**

1. Ознайомитись з теоретичним матеріалом.

2. В обраному середовищі програмування створити віконний проект та написати програму, яка реалізує алгоритм сортування підрахунком.

3. Реалізувати варіант згідно списку підгрупи, додавши 5.

4. Передбачити вивід у вікні вхідних даних, результатів виконання варіанту, проміжних результатів згідно алгоритму та результуючий відсортований масив даних.

5. Оформити звіт про виконання лабораторної роботи.

**Індивідуальне завдання(варіант 9)**

Задано одномірний масив дійсних чисел. Впорядкувати елементи, розташовані між першим і останнім від'ємним елементом по зростанню.

**Теоретичні відомості**

Сортування підрахунком – алогритм впорядкування, що застосовується при малій кількості різних елементів у масиві даних. Складність: O(n + k). Суть в тому, що ми створюємо новий масив, кілкість елементів якого дорівнює діапазону чисел першого масиву. Після цього ми просто рахуємо кілкість елементів у нашому основному масиві, які дорівнюють <індекс підмасиву + мінімальний елемент основного масиву>.

**Уривок коду програми**

**Індивідуальне завдання**

void CountingSort::IndividualTask()

{

size\_t startIndex;

for (size\_t i = 0; i < size; i++)

{

if (arr[i] < 0)

{

startIndex = i;

break;

}

}

size\_t endIndex;

for (size\_t i = size - 1; i >= 0; i--)

{

if (arr[i] < 0)

{

endIndex = i;

break;

}

}

size\_t newSize = size - (size - endIndex - 1) - startIndex;

size\_t checker = 0;

int\* tmp = new int[newSize];

for (size\_t i = 0; i <= endIndex; i++)

{

if (i < startIndex)

{

checker++;

continue;

}

tmp[i - checker] = arr[i];

}

size = newSize;

delete[]arr;

arr = tmp;

}

**Алгоритм сортування підрахунком**

void CountingSort::Sorting()

{

int min = arr[0];

int max = arr[0];

size\_t numberToAdd;

for (size\_t i = 0; i < size; i++)

{

if (arr[i] < min)

min = arr[i];

}

if (min < 0)

{

for (size\_t i = 0; i < size; i++)

{

arr[i] -= min;

}

numberToAdd = -min;

min -= min;

}

for (size\_t i = 0; i < size; i++)

{

if (arr[i] > max)

max = arr[i];

}

sizeOfSecondArr = max - min + 1;

countOfNumbers = new size\_t[sizeOfSecondArr];

size\_t k = 0;

int\* tmp = new int[size];

for (size\_t i = 0; i < sizeOfSecondArr; i++)

{

countOfNumbers[i] = 0;

for (size\_t j = 0; j < size; j++)

if (arr[j] == min)

countOfNumbers[i]++;

if (countOfNumbers[i] != 0)

{

for (size\_t j = 0; j < countOfNumbers[i]; j++)

{

tmp[k] = min;

k++;

}

}

min++;

}

delete[]arr;

arr = tmp;

for (size\_t i = 0; i < size; i++)

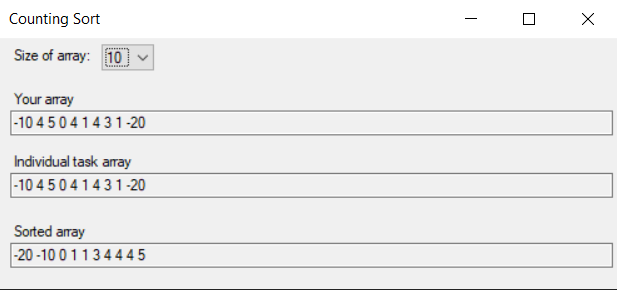
{

arr[i] -= numberToAdd;

}

}

**Результат виконання програми**

****

**Висновок**

Виконуючи дану лабораторну роботу, я вивчив алгоритм сортування підрахунком, реалізував його в програмі та дослідив його швидкодію.