Додаток 1

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

Звіт

з лабораторної роботи №7
з дисципліни «Алгоритми та структури даних-1. Основи алгоритмізації»
«Дослідження лінійного пошуку в послідовностях»
Варіант 6

Виконав студент ІП-15, Волинець Кирило Михайлович
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Лабораторна робота 7

Дослідження лінійного пошуку в послідовностях

Мета - дослідити методи послідовного пошуку у впорядкованих і невпорядкованих послідовностях та набути практичних навичок їх використання під час складання програмних специфікацій.

Індивідуальне завдання

Варіант 6

6	73 - i	64 + 2 * i	Кількість елементів між максимальним та мінімальним елементами

Побудова математичної моделі.

Складемо таблицю імен .

Змінна	***Тип***	***Ім'я***	***Призначення***
Розмір масивів (=10)	Цілочисельний	array_size	Стала
Перший масив	Символьний[array_size]	array1	Вхідні
Другий масив	Символьний[array_size]	array2	Вхідні
Масив рівних елементів	Символьний[array_size]	array3	Проміжні
Ітератор1	Цілочисельний	i	Проміжні
Ітератор2	Цілочисельний	j	Проміжні
Розмір третього масиву	Цілочисельний	array3_size	Проміжні
Побудова першого і другого масивів	Пустий (функція)	build_arrays(char*, char*)	Обчислення
Знаходження рівних елементів та їх кількості	Цілочисельний (функція)	find_equal_elements(char*, char*, char*)	Обчислення
Знаходження кількості елементів між максимальним та мінімальним	Цілочисельний (функція)	distance_between_max_min(char*, int)	Обчислення
Індекс мінімального	Цілочисельний	min_i	Проміжні
Індекс максимального	Цілочисельний	max_i	Проміжні
Модуль	Цілочисельний (функція)	abs()	Обчислення

Побудуємо два масиви array1 та array2 за допомогою циклу та ітератору. Обчислення будуть здійснені в функції build_arrays. Дані виводяться за допомогою покажчиків. Перший масив буде відсортований за спаданням, а другий за зростанням, тому що вони задані арифметичною прогресією. Знайдемо однакові елементи шляхом порівнювання кожного з кожним. Потім знайдемо відстань між максимальним та мінімальним за допомогою distance_between_max_min. Обійшовши масив, знайдемо позиції максимального та мінімального max_p min_p, та віднімемо їх. Треба взяти модуль, бо відстань від'ємна та відняти ще одиницю для правильного результату.

Розв'язання

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.

Псевдокод

Основна програма:

початок
    build_arrays(array1, array2);
    array3_size = find_equal_elements(array1, array2, array3);
    вивести distance_between_max_min(array3, array3_size)
кінець

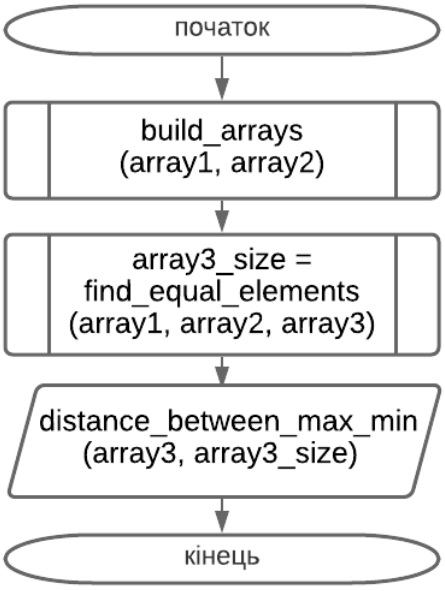
Підпрограми:

build_arrays(char* array1, char* array2)
    повторити для i від 0 до i < array_size:
        array1[i] = 73 - i;
        array2[i] = 64 + 2 * i;
    все повторити
кінець build_arrays

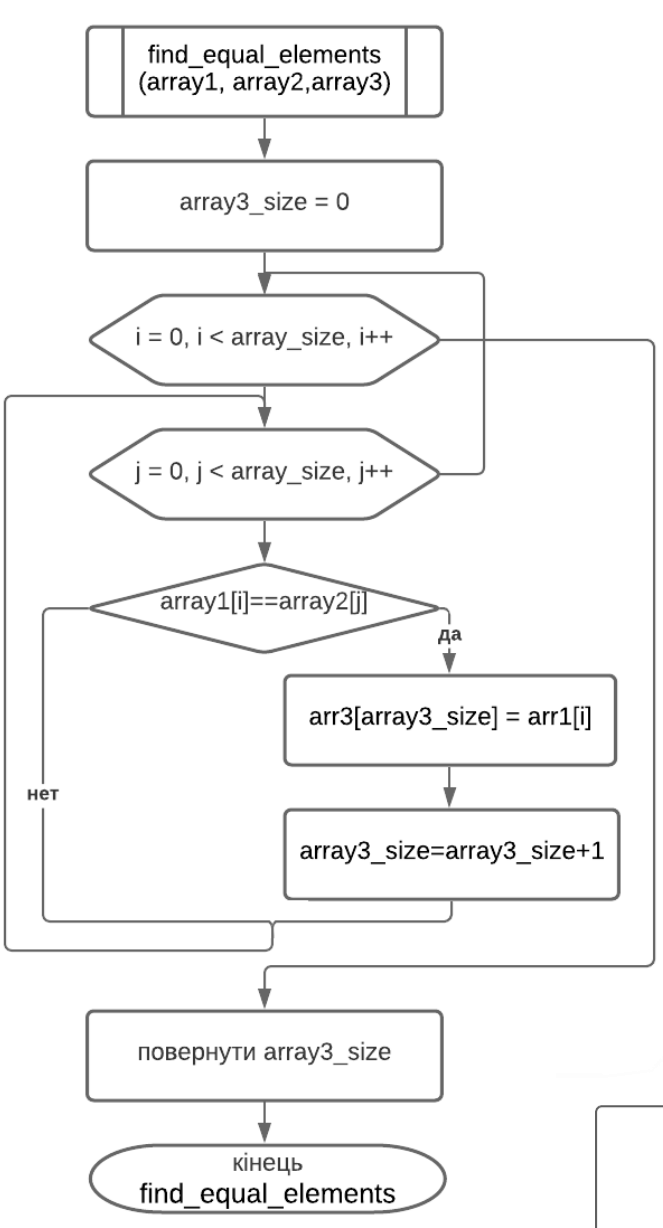
find_equal_elements(char* arr1, char* arr2, char* arr3)
    array3_size = 0
    повторити для i від 0 до arr_size:
        повторити для j від 0 до arr_size:
            якщо arr1[i] == arr2[j] то:
                arr3[array3_size] = arr1[i]
                array3_size = array3_size + 1
        все повторити
    все повторити
    повернути array3_size
кінець find_equal_elements

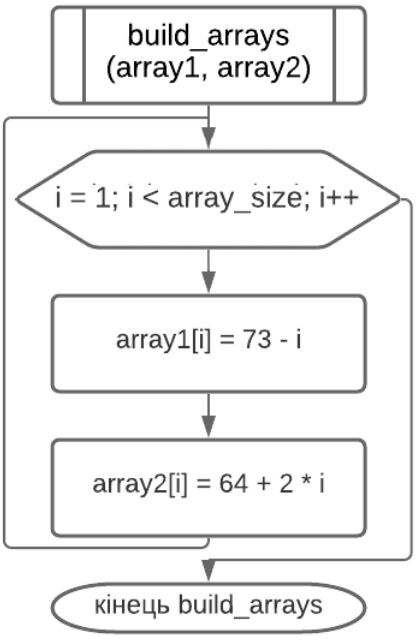
distance_between_max_min(char* array3, int array3_size)
    max_i = 0
    min_i = 0
    повторити для i від 1 до i < array3_size:
        якщо array3[i] > array3[min_i]:
            max_i = i
        все якщо
        інакше якщо array3[i] < array3[min_i]:
            min_i = i
        все якщо
    все повторити
    повернути abs(max_i - min_i) - 1
кінець distance_between_max_min

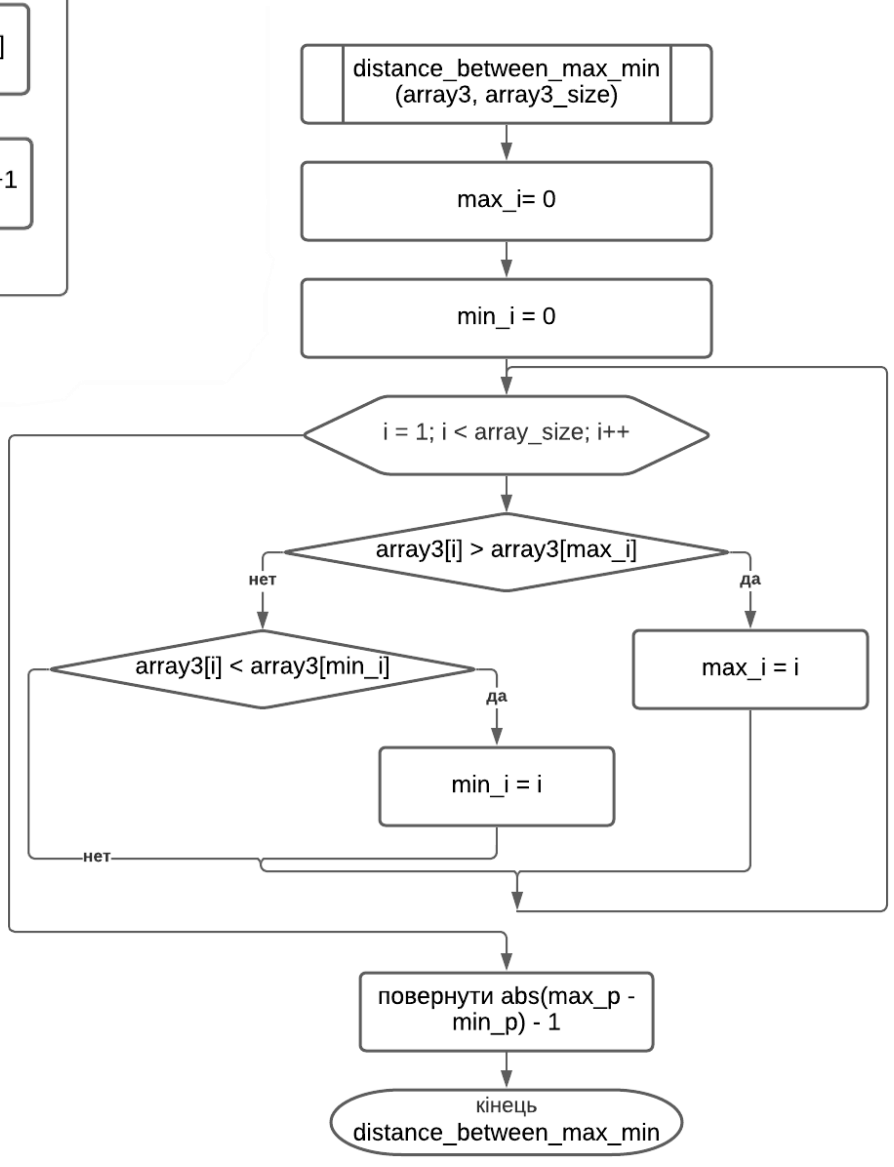
Блок-схема:



Підпрограми:


find_equal_elements
(array1, array2,array3)
array3_size = 0
i = 0, i < array_size, i++
j = 0, j < array_size, j++
array1[i]==array2[j]
да
нет
arr3[array3_size] = arr1[i]
array3_size=array3_size+1
повернути array3_size
кінець
find_equal_elements


build_arrays
(array1, array2)
i = 1; i < array_size; i++
array1[i] = 73 - i
array2[i] = 64 + 2 * i
кінець build_arrays


distance_between_max_min
(array3, array3_size)
max_i= 0
min_i = 0
i = 1; i < array_size; i++
array3[i] > array3[max_i]
нет
да
array3[i] < array3[min_i]
да
max_i = i
min_i = i
нет
повернути abs(max_p -
min_p) - 1
кінець
distance_between_max_min

Код програми

```cpp
#include <iostream>
using namespace std;

const int arr_size = 10;

void build_arrays(char*, char*);
int find_equal_elements(char*, char*, char*);
int distance_between_max_min(char*, int);

int main()
{
    setlocale(0, "");
    char array1[arr_size], array2[arr_size];
    build_arrays(array1, array2);
    cout << "Перший масив:" << endl;
    for (int i = 0; i < arr_size; i++) {
        cout << array1[i] << "(" << int(array1[i]) << ") ";
    }
    cout << endl << "Другий масив:" << endl;
    for (int i = 0; i < arr_size; i++) {
        cout << array2[i] << "(" << int(array2[i]) << ") ";
    }
    char array3[arr_size];
    int array3_size = find_equal_elements(array1, array2, array3);
    cout << endl << "Третій масив:" << endl;
    for (int i = 0; i < array3_size; i++) {
        cout << array3[i] << "(" << int(array3[i]) << ") ";
    }
    cout << endl << "Кількість елементів між мінімальним та максимальним = " << distance_between_max_min(array3, array3_size);
}

void build_arrays(char* arr1, char* arr2) {
    for (int i = 0; i < arr_size; i++) {
        arr1[i] = 73 - i;
        arr2[i] = 64 + 2 * i;
    }
}
int find_equal_elements(char* arr1, char* arr2, char* arr3) {
    int array3_size = 0;
    for (int i = 0; i < arr_size; i++) {
        for (int j = 0; j < arr_size; j++) {
            if (arr1[i] == arr2[j]) {
                arr3[array3_size] = arr1[i];
                array3_size++;
            }
        }
    }
    return array3_size;
}

int distance_between_max_min(char* array3, int array3_size) {
    int max_i = 0, min_i = 0;
    for (int i = 1; i < array3_size; i++) {
        if (array3[i] > array3[max_i]) {
            max_i = i;
        }
        else if (array3[i] < array3[min_i]) {
            min_i = i;
        }
    }
    return abs(max_i - min_i) - 1;
}
```

Випробування коду

```
Перший масив:
I(73) H(72) G(71) F(70) E(69) D(68) C(67) B(66) A(65) @(64)
Другий масив:
@(64) B(66) D(68) F(70) H(72) J(74) L(76) N(78) P(80) R(82)
Третій масив:
@(64) B(66) D(68) F(70) H(72)
Кількість елементів між мінімальним та максимальним = 3
C:\Users\kiril\source\repos\ASD7\Debug\ASD7.exe (процесс 24808) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "А
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно…
```

Висновки

Ми дослідили методи послідовного пошуку у впорядкованих і невпорядкованих послідовностях та набули практичних навичок їх використання під час складання програмних специфікацій. В результаті виконання лабораторної роботи ми отримали алгоритм для визначення двох масивів за формулою та знаходження кількості елементів між максимальним та мінімальним значень серед спільних членів обох масивів. В процесі випробування програма вивела три масиви та відповідь, що є вірною.