Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний

інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 8 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

«Дослідження алгоритмів пошуку та сортування»

Варіант 5

Виконав студент ІП-12, Василишин Михайло Михайлович

(шифр, прізвище, ім'я, по батькові)

Перевірив Василишин Михайло Михайлович

( прізвище, ім'я, по батькові)

Київ 2021

**Лабораторна робота №8** «Дослідження лінійного пошуку в послідовностях»

**Мета** – дослідити алгоритми пошуку та сортування, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

**Варіант** – 5

**Задача №5**.

Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (табл. 1).

2. Ініціювання змінної, що описана в п.1 даного завдання.

3. Створення нової змінної індексованого типу (одновимірний масив) та її ініціювання значеннями, що обчислюються згідно з варіантом (табл. 1).

|  |  |  |  |
| --- | --- | --- | --- |
| № | Розмірність | Тип даних | Обчислення масиву |
| 5 | 5\*7 | Цілий | Із суми значень елементів стовпців двовимірного масиву. Відсортувати методом бульбашки за зростанням. |

**Розв’язок**

1. **Постановка задачі.**

Результатом розв’язку є змінна індексованого типу, що відповідає одновимірному масиву. Потрібні проміжні змінні для подальших обчислень задані в умові задачі, а деякі – генеруватимуться випадково. Для визначення результату будуть використані проміжні змінні та допоміжні функції. Початкових даних для розв’язку не потрібно.

1. **Побудова математичної моделі.** Складемо таблицю імен змінних.

|  |  |  |  |
| --- | --- | --- | --- |
| Змінна | Тип | Ім’я | Призначення |
| Початковий масив | Цілий двовимірний масив | Array2D | Проміжне дане |
| Результуючий масив | Цілий масив | Array1D | Проміжне дане |
| Лічильник для циклу | Цілий | i | Проміжне дане |
| Лічильник для циклу | Цілий | j | Проміжне дане |
| Сума по стовпцю | Цілий | sum | Проміжне дане |
| Змінна для обміну | Цілий | temp | Проміжне дане |

Опис розв’язання слід почати з деталізації початкових даних, маємо: розмірність масиву array2D – 5\*7, самі ж значення елементів будуть генеруватись випадково. Отже, для оперування першою структурою даних потрібно визначити дві допоміжні функції: generate2D(), та output2D().

Отримавши масив array2D, ініціалізуємо одновимірний масив array1D, кожен елемент якого дорівнює відповідно сумі по стовпцю масиву array2D. Для нього теж виділено дві допоміжні функції: generate1D(), output1D().

Після ініціалізації деякими значеннями масиву array1D, потрібно відсортувати його методом бульбашки. Сам метод є базовим алгоритмом сортування, який легко описати. Будемо використовуючи для нього функцію sort().

Отож, розв’язок зводиться до ініціалізації двох масивів та сортування другого одновимірного масиву. Слід звернути увагу на те, що після кожної модифікації алгоритму його вміст виводитеметься.

**Крок 1.** Визначимо основні дії.

**Крок 2.** Деталізуємо функції generate2D та output2D

**Крок 3.** Деталізуємо функції generate1D та output1D

**Крок 4.** Деталізуємо функцію sort()

1. **Псевдокод алгоритму**

Крок 1

**початок**

generate2D(array2D,m,n)

output2D((array2D,m,n)

generate1D(array1D, array2D ,m,n)

output1D((array1D,n)

sort(array1D,n)

output1D((array1D,n)

**Кінець**

**функція** generate2D()

**функція** output2D()

**функція** generate1D()

**функція** output1D()

**функція** sort()

Крок 2

**початок**

generate2D(array2D,m,n)

output2D((array2D,m,n)

generate1D(array1D, array2D ,m,n)

output1D((array1D,n)

sort(array1D,n)

output1D((array1D,n)

**Кінець**

**Початок функції** generate2D(array2D[][],m,n)

**повторити**

**для** і **від** 0 **до** m

**повторити**

**для** j **від** 0 **до** n

array2D[i+1,j+1]:= rand()%1000

**все повторити**

**все повторити**

**кінець функції** generate2D(array2D[][],size)

**Початок функції** output2D(array2D[][],m,n)

**повторити**

**для** і **від** 0 **до** m

**повторити**

**для** j **від** 0 **до** n

**вивід** array2d[I+1,j+1]

**все повторити**

**все повторити**

**кінець функції** output2D (array2D[][],size)

**функція** generate1D()

**функція** output1D()

**функція** sort()

Крок 3

**початок**

generate2D(array2D,m,n)

output2D((array2D,m,n)

generate1D(array1D, array2D ,m,n)

output1D((array1D,n)

sort(array1D,n)

output1D((array1D,n)

**Кінець**

**Початок функції** generate2D(array2D[][],m,n)

**повторити**

**для** і **від** 0 **до** m

**повторити**

**для** j **від** 0 **до** n

array2D[i+1,j+1]:= rand()%1000

**все повторити**

**все повторити**

**кінець функції** generate2D(array2D[][],size)

**Початок функції** output2D(array2D[][],m,n)

**повторити**

**для** і **від** 0 **до** m

**повторити**

**для** j **від** 0 **до** n

**вивід** array2d[I+1,j+1]

**все повторити**

**все повторити**

**кінець функції** output2D (array2D[][],size)

**Початок функції** generate1D(array1D[],array2D[][],m,n)

**повторити**

**для** j **від** 0 **до** n

sum:=0

**повторити**

**для** i **від** 0 **до** m

sum = sum+array2D[i+1,j+1]

**все повторити**

array1D[j+1] := sum

**все повторити**

**кінець функції**  generate1D (array1D[],array2D[][],m,n)

**Початок функції** output1D(array1D[],n)

**повторити**

**для** і **від** 0 **до** n

**вивід** array1D[i+1]

**все повторити**

**кінець функції** output1D(array1D[],n)

**функція** sort()

Крок 4.

**початок**

generate2D(array2D,m,n)

output2D((array2D,m,n)

generate1D(array1D, array2D ,m,n)

output1D((array1D,n)

sort(array1D,n)

output1D((array1D,n)

**Кінець**

**Початок функції** generate2D(array2D[][],m,n)

**повторити**

**для** і **від** 0 **до** m

**повторити**

**для** j **від** 0 **до** n

array2D[i+1,j+1]:= rand()%1000

**все повторити**

**все повторити**

**кінець функції** generate2D(array2D[][],size)

**Початок функції** output2D(array2D[][],m,n)

**повторити**

**для** і **від** 0 **до** m

**повторити**

**для** j **від** 0 **до** n

**вивід** array2d[I+1,j+1]

**все повторити**

**все повторити**

**кінець функції** output2D (array2D[][],size)

**Початок функції** generate1D(array1D[],array2D[][],m,n)

**повторити**

**для** j **від** 0 **до** n

sum:=0

**повторити**

**для** i **від** 0 **до** m

sum = sum+array2D[i+1,j+1]

**все повторити**

array1D[j+1] := sum

**все повторити**

**кінець функції**  generate1D (array1D[],array2D[][],m,n)

**Початок функції** output1D(array1D[],n)

**повторити**

**для** і **від** 0 **до** n

**вивід** array1D[i+1]

**все повторити**

**кінець функції** output1D(array1D[],n)

**Початок функції** sort(array1D[],n)

**повторити**

**для** і **від** 0 **до** n

**повторити**

**для** і **від** 0 **до** n

**якщо** array1D[i+2] < array1D[i+1]

**то**

temp = array1D[i+2]

array1D[i+2] = array1D[i+1]

array1D[i+1] = temp

**все якщо**

**все повторити**

**все повторити**

**кінець функції** sort(array1D[],n)

1. **Блок-схема**

Крок 1



Крок 2.



Крок 3.



Крок 4.



1. **Код програми(с++)**

**#include<iostream>**

**#include<time.h>**

**#include<iomanip>**

**using namespace std;**

**void generate2D(int \*\*array2D, int m,int n)**

**{**

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

**for(int j = 0; j < n;j++)array2D[i][j] = rand()%1000;**

**}**

**void output2D(int \*\*array2D, int m,int n)**

**{**

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

**{**

**for(int j = 0; j < n;j++)cout <<setw(4)<< array2D[i][j];**

**cout << endl;**

**}**

**cout << endl;**

**}**

**void generate1D(int \*array1D,int \*\*array2D,int m,int n)**

**{**

**for(int j = 0; j < n;j++)**

**{**

**int sum = 0;**

**for(int i = 0; i < m;i++)sum+=array2D[i][j];**

**array1D[j] = sum;**

**}**

**}**

**void output1D(int \*array1D,int n)**

**{**

**for(int i = 0; i < n;i++)cout << array1D[i] << " ";**

**cout << endl << endl;**

**}**

**void sort(int\* array1D, int n)**

**{**

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

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

**if(array1D[j+1] < array1D[j])**

**{**

**int temp = array1D[j+1];**

**array1D[j+1] = array1D[j];**

**array1D[j] = temp;**

**}**

**}**

**int main()**

**{**

**srand(time(NULL));**

**int m = 5,n = 7;**

**int \*array1D = new int[n];**

**int \*\*array2D = new int\*[m];**

**for(int i = 0; i < m;i++)array2D[i] = new int[n];**

**generate2D(array2D,m,n);**

**output2D(array2D,m,n);**

**generate1D(array1D,array2D,m,n);**

**output1D(array1D,n);**

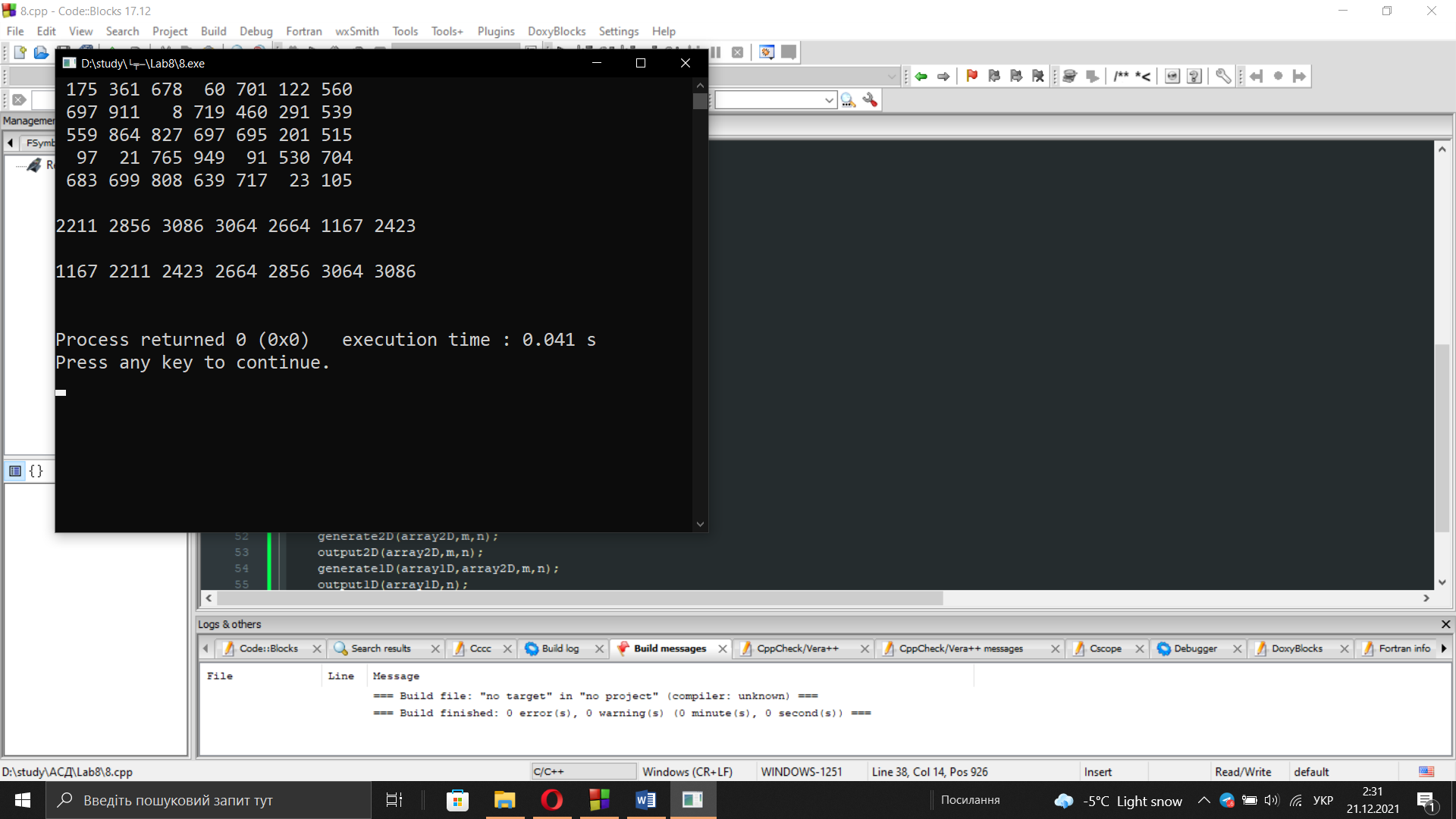
**sort(array1D,n);**

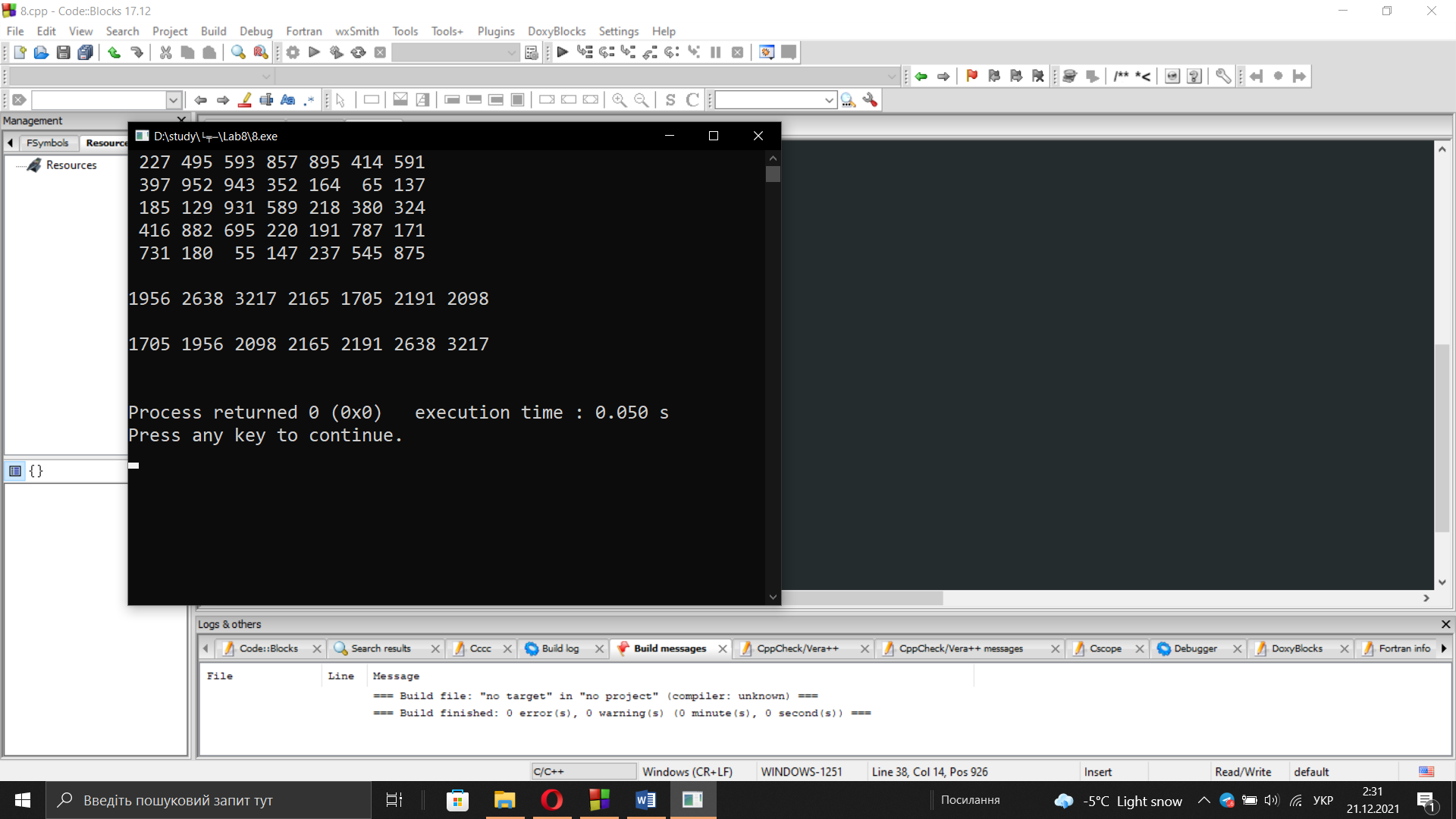
**output1D(array1D,n);**

**return 0;**

**}**

1. **Тестування програми**





1. Висновок.

На даній лабораторній роботі було досліджено алгоритми пошуку та сортування, набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій.. Особливістю моєї роботи було використання спеціалізованих функцій для ініціалізації масивів, а також реалізація одного із базових алгоритмів сортування. Також особливістю, було визначення одновимірного масиву через двовимірний за за принципом суми елементів стовпця.