## МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬ­НИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Інститут **КНІТ**

Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи № 3

**На тему:** *“* *Робота з масивами в С ”*

**З дисципліни:** *“Основи програмування”*

## Лектор:

ст.викл. каф. ПЗ

Муха Т.О.

**Виконав:**

ст. гр. ПЗ-18

Юшкевич А.І.

**Прийняв:**

ст. викл. каф. ПЗ

Кутельмах. Р.К.

« » 2022 р.

∑= .

Львів – 2022

**Тема роботи:** робота з масивами в C.

**Мета роботи:** навчитися організовувати такі структури даних як масиви та освоїти основні методи програмування алгоритмів обробки масивів даних засобами мови С.

**Індивідуальне завдання**

**1)**

Написати програму для обробки даних, організованих у масив, згідно завдання наведеного варіанту. У програмі використати такі форми звертання до елементів масиву:

* на основі індексів,
* через розадресоване ім’я масиву і зміщення елемента.

**25.** Ввести вектор дійсних чисел. Всі елементи вектора, що розмішені перед його мінімальним елементом, зменшити на половину від середнього арифметичного елементів введеного вектора.

**2)**

Написати програму для обробки даних, організованих у масив, згідно завдання наведеного варіанту. У програмах використати різні форми звертання до елементів багатовимірних масивів, у тому числі за допомогою конструкцій “вказівник на масив” і “вказівник на вказівник”. Враховувати, що кожен рядок матриці можна опрацьовувати як окремий елемент.

**25.** int A[15][15], В[15][15]; Знайти С=АВ.

**Теоретичні відомості**

Масив являє собою сукупність однотипних змінних, розміщених у послідовно пронумерованих суміжних комірках пам'яті. Номер елемента масиву задається індексом. ***Індексація елементів масиву в С починається з 0***. Якщо у масиві N елементів, то перший елемент матиме індекс 0, а останній – індекс (N–1).

За способом зв'язування індексів з комірками пам'яті виділяють три категорії масивів:

*статичні*, *фіксовані автоматичні* та *динамічні*.

*Статичним* називають масив, в якому зв’язування індексів та розміщення в пам'яті виконуються на етапі компіляції програми. Статичні масиви досить ефективні, оскільки для їх створення та знищення не потрібно додаткових операцій.

*Фіксованим автоматичним* називають масив, в якому індекси зв'язуються статично, а розміщення в пам'яті виконується при обробці оголошень всередині функцій.

*Динамічним* називають масив, де зв’язування індексів і розміщення в пам'яті виконується безпосередньо під час виконання програми. Цей спосіб організації даних достатньо гнучкий, оскільки розміри динамічного масиву можуть збільшуватися та зменшуватися під час виконання програми у міру необхідності.

Об’єм пам'яті, який займає масив, рівний добутку розміру типу елементів масиву та кількості елементів в масиві. Наприклад, якщо масив містить 16 чисел типу int і тип int займає в пам’яті 2 байти, то цілий масив займатиме 2×16 = 32 байти.

***У мові С не перевіряється вихід індексу масиву за межі допустимого значення***. Вихід за межі пам’яті, яку займає масив, не контролюється. При помилці програміста може відбутися запис інформації в сусідні комірки пам'яті, а ***наслідки*** цього ***непередбачувані***.

Масиви можуть бути одновимірними та багатовимірними.

**Одновимірні масиви**

Одновимірні масиви оголошуються таким чином:

тип ім’я\_масиву[розмір]

Наприклад, так оголошується статичний масив з іменем numbs, що складається з 10

елементів типу int:

int numbs[10];

У С доступ до елементів масиву здійснюється за допомогою спеціального оператора []. В квадратних дужках вказується індекс елемента масиву, а перед ними – його ім'я. Наприклад, присвоєння першому елементу масиву numbs значення 12 виконується так:

numbs[0] = 12;

Масиви тісно пов’язані із вказівниками. Ім’я масиву є вказівником на його перший елемент.

Тому ім'я масиву можна використати в якості бази для зміщення вказівника. Наприклад, вираз

numbs[4] = 1

еквівалентний виразу

\*(numbs+4) = 1;

**Ініціалізація масивів**

Для заповнення масиву початковими даними достатньо виконати ініціалізацію під час оголошення масиву:

int numbs[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

При цьому не потрібно вказувати розмір масиву – він буде розпізнаний за кількістю введених елементів. Тому можна писати так:

int numbs[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

Більше того, якщо вказати в квадратних дужках неправильний розмір, все одно компілятор автоматично обчислить правильний розмір. Наприклад, при оголошенні:

int numbs[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8}

компілятор «вважатиме», що масив містить 9 елементів. Однак, при цьому виникне проблема при наступному присвоєнні:

numbs[10] = 10;

Компілятор розпізнає помилку, коли присвоюється більше чисел, ніж задано елементів масиву:

int numbs[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

При цьому видається повідомлення «Занадто багато ініціалізаторів».

До масивів застосовний оператор **sizeof**, який повертає розмір масиву у байтах:

sizeof(numbs)

Якщо масив не підлягає змінам, його можна оголосити константним, додавши слово **const** в оголошення масиву:

const int numbs[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

**Двовимірні масиви**

Часто за допомогою мов програмування виникає потреба обробляти таблиці або матриці.

Для їх представлення у мові С передбачені багатовимірні масиви.

Найпоширеніший варіант багатовимірного масиву – двовимірний масив, який можна представити у вигляді масиву одновимірних масивів:

тип ім’я\_масиву [розмір1][розмір2]

Звертаються до елементів масиву так:

ім’я\_масиву [індекс1][індекс2]

Наприклад:

int matr[10][5]; */\* оголошується матриця з 10 рядків і 5 стовпців \*/*

matr[4][3]=10; */\* четвертому елементу п’ятого рядка присвоюється значення 10 \*/*

Ініціалізація двовимірного масиву здійснюється по рядках:

int numbs[3][3] = {1, 2, 3, 4, 5, 6, 7, 8, 9};

В результаті перший рядок міститиме числа 1, 2, 3, другий – числа 4, 5, 6, а третій – числа

7, 8, 9.

Для наочності краще при ініціалізації використовувати групування:

int numbs[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

Двовимірний масив, як і будь-який інший, зберігається в послідовних комірках пам'яті, а тому його можна представити як одномірний, пам’ятаючи, що ім'я масиву – це вказівник на його перший елемент. Нехай двовимірний масив mas має розмірність n\*m. Тоді елемент mas[i][j] можна представити як \*(\*mas+n\*i+j). З іншого боку, вираз mas[i] є вказівником на перший елемент і-ого рядка, а тому \*[mas[i]+j] – це j-ий елемент і-ого рядка.

**№1**

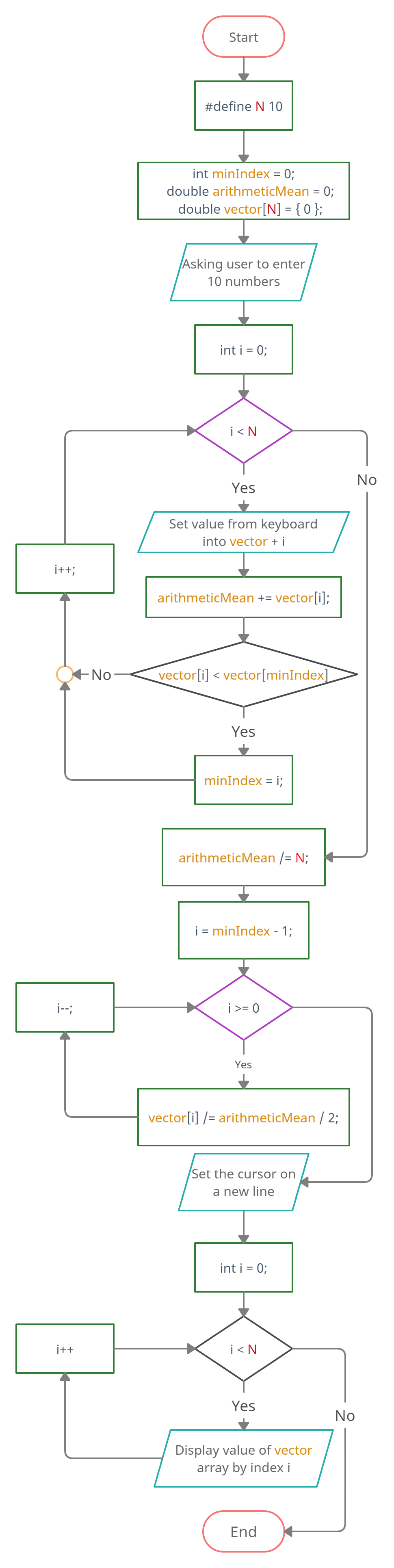


Рис.1 Блок-схема першої програми

# Код програми

Назва файлу: **Lab03.Dod1.c**

#include <stdio.h>

#include <math.h>

#define N 10

int main(void) {

int minIndex = 0;

double arithmeticMean = 0;

double vector[N] = { 0 };

printf("Enter 10 numbers:\n");

for (int i = 0; i < N; i++) {

scanf\_s("%lf", vector + i);

arithmeticMean += vector[i];

if (vector[i] < vector[minIndex])

minIndex = i;

}

arithmeticMean /= N;

for (int i = minIndex - 1; i >= 0; i--) {

vector[i] /= arithmeticMean / 2;

}

printf("\n");

for (int i = 0; i < N; i++) {

printf("%.3lf\t", \*(vector + i));

}

return 0;

}

# Протокол роботи

1. Користувач вводить 10 чисел
2. Програма виводить введений вектор чисел в консоль так, що всі елементи вектора, які розмішені перед його мінімальним елементом, зменшені на половину від середнього арифметичного елементів введеного вектора.

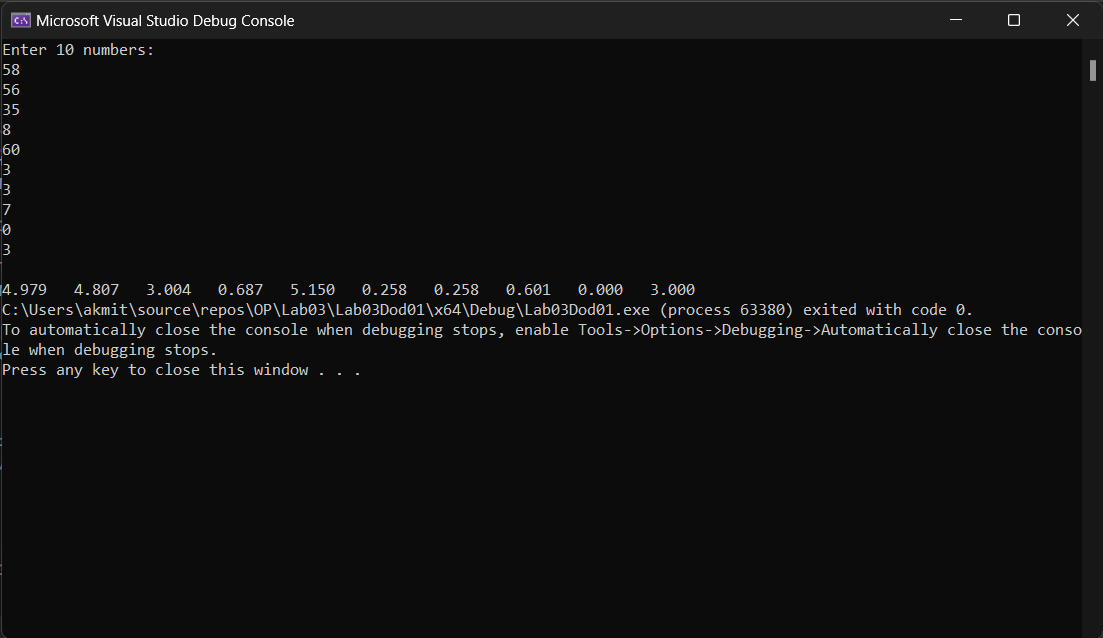


Рис.2 Результат виконання першої програми.

**№2**

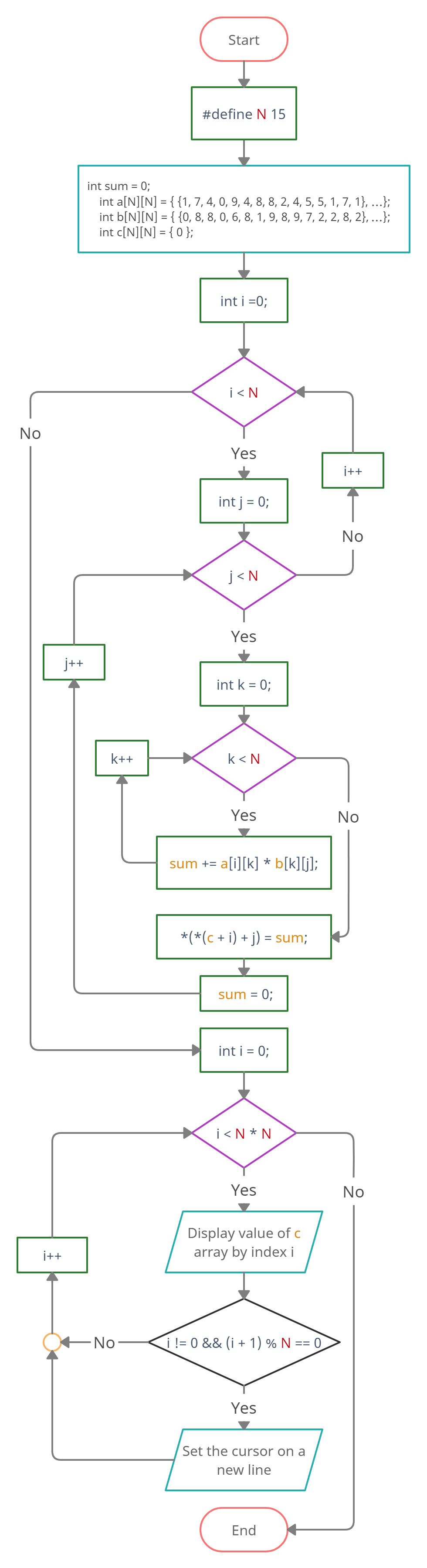


Рис.3 Блок-схема другої програми.

**Код програми**

Назва файлу: **Lab03.Dod2.c**

#include <stdio.h>

#define N 15

int main(void) {

int sum = 0;

int a[N][N] = { {1, 7, 4, 0, 9, 4, 8, 8, 2, 4, 5, 5, 1, 7, 1},

{1, 5, 2, 7, 6, 1, 4, 2, 3, 2, 2, 1, 6, 8, 5},

{7, 6, 1, 8, 9, 2, 7, 9, 5, 4, 3, 1, 2, 3, 3},

{4, 1, 1, 3, 8, 7, 4, 2, 7, 7, 9, 3, 1, 9, 8},

{6, 5, 0, 2, 8, 6, 0, 2, 4, 8, 6, 5, 0, 9, 0},

{0, 6, 1, 3, 8, 9, 3, 4, 4, 6, 0, 6, 6, 1, 8},

{4, 9, 6, 3, 7, 8, 8, 2, 9, 1, 3, 5, 9, 8, 4},

{0, 7, 6, 3, 6, 1, 5, 4, 2, 0, 9, 7, 3, 7, 2},

{6, 0, 1, 6, 5, 7, 5, 4, 1, 2, 0, 0, 1, 4, 6},

{0, 7, 1, 7, 7, 7, 7, 3, 3, 5, 9, 9, 8, 1, 8},

{2, 6, 6, 0, 3, 8, 0, 1, 2, 5, 0, 9, 4, 7, 8},

{3, 5, 1, 2, 0, 1, 6, 4, 0, 6, 1, 8, 9, 8, 4},

{1, 4, 3, 9, 8, 8, 0, 8, 7, 7, 8, 3, 8, 3, 7},

{1, 0, 7, 3, 4, 9, 6, 5, 1, 0, 9, 9, 6, 8, 3},

{4, 8, 4, 9, 9, 2, 5, 5, 3, 3, 3, 7, 4, 3, 8} };

int b[N][N] = { {0, 8, 8, 0, 6, 8, 1, 9, 8, 9, 7, 2, 2, 8, 2},

{8, 9, 0, 7, 8, 1, 5, 8, 6, 1, 2, 4, 2, 5, 8},

{6, 2, 6, 5, 3, 9, 2, 4, 6, 1, 8, 2, 1, 1, 9},

{7, 6, 2, 9, 5, 2, 0, 0, 3, 9, 1, 8, 1, 9, 5},

{3, 2, 5, 2, 5, 8, 6, 7, 7, 2, 2, 9, 4, 1, 9},

{6, 9, 8, 2, 5, 5, 4, 9, 1, 2, 5, 0, 8, 3, 9},

{3, 9, 6, 7, 9, 9, 7, 6, 9, 3, 5, 7, 6, 6, 5},

{8, 2, 5, 4, 4, 1, 6, 1, 6, 3, 3, 5, 5, 3, 2},

{8, 2, 5, 3, 6, 1, 8, 6, 2, 1, 4, 6, 2, 9, 1},

{5, 0, 3, 6, 4, 9, 2, 9, 3, 4, 4, 0, 5, 9, 6},

{3, 4, 2, 8, 8, 7, 5, 8, 1, 2, 5, 7, 4, 4, 4},

{4, 4, 2, 9, 3, 0, 7, 7, 8, 7, 3, 0, 3, 8, 2},

{5, 2, 0, 3, 7, 2, 9, 7, 5, 3, 4, 3, 2, 1, 1},

{3, 2, 5, 3, 5, 1, 2, 2, 2, 6, 4, 0, 7, 0, 3},

{4, 0, 0, 5, 3, 1, 0, 6, 7, 5, 0, 0, 4, 5, 4}, };

int c[N][N] = { 0 };

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

for (int k = 0; k < N; k++)

sum += a[i][k] \* b[k][j];

\*(\*(c + i) + j) = sum;

sum = 0;

}

}

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

printf("%d\t", \*(\*c + i));

if (i != 0 && (i + 1) % N == 0)

printf("\n");

}

return 0;

}

**Протокол виконання**

1. Програма виводить результат множення двох заданих матриць розмірністю 15\*15

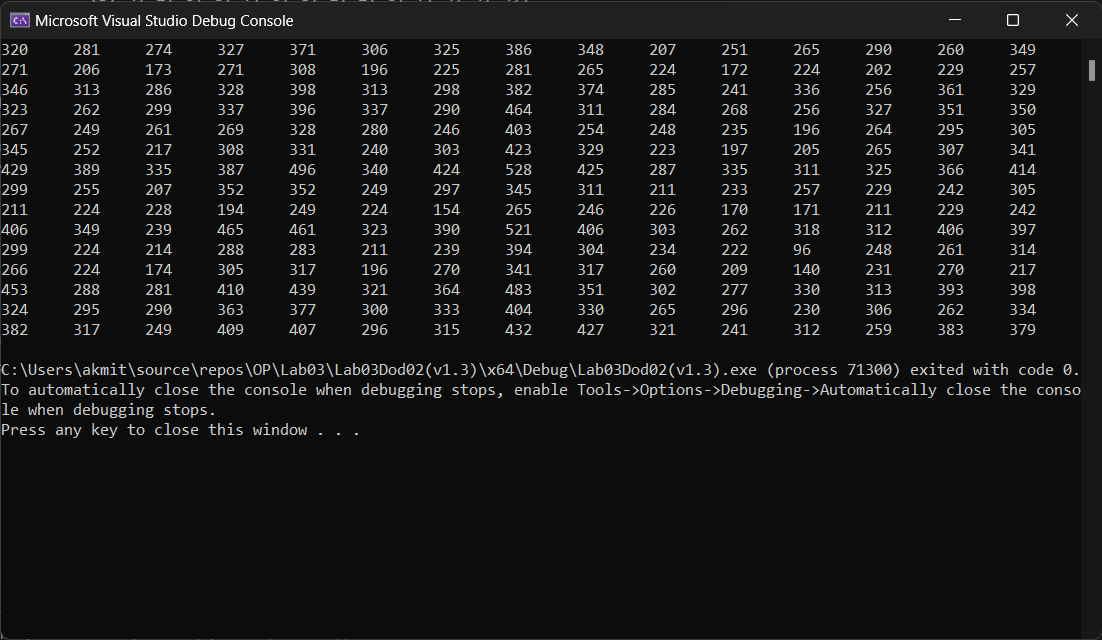


Рис.4 Результат виконання другої програми.

**Висновок**

На лабораторній роботі ми навчитися організовувати такі структури даних як масиви та освоїли основні методи програмування алгоритмів обробки масивів даних засобами мови С. Розібралися з таким поняттям як вказівники та навчилися правильно їх застосовувати.