**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-14

Губик А. С.

**Прийняв:**

ст.викл. каф ПЗ

Кутельмах Р. К.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑= \_\_\_\_ .

Львів – 2022

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

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

**Індивідуальне завдання**

Використовуючи цикл:

Додаток 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. Перевірити, чи введений з клавіатури вектор дійсних чисел є спадним. Якщо ні – то визначити найменший елемент введеного вектора і поміняти його місцями з останнім.

Додаток 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.Ввести прямокутну матрицю цілих чисел. Циклічно зсунути її рядки на 2 вгору. Циклічний зсув передбачає, що елементи двох перших рядків матриці заносяться на відповідні місця зсунутих елементів двох останніх рядків.

**Теоретичні відомості**

**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; 3

**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: 6

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.

**Код програми**

Додаток 1.

Назва файлу: **vec.c**

#include <stdio.h>

int main(){

int VecLength = 0;

float vec[100] = {0};

for(;;){

int n;

printf("Enter a real number or something else to stop: ");

n = scanf("%f", &vec[VecLength]);

if (!n){

printf("------------------------------------------------\n");

break;

}

VecLength ++;

}

if (VecLength == 0){

printf("Vector is empty\n");

return 0;

}

int flag = 1;

for(int i = 0; i < VecLength - 1; i++)

if (\*(vec + i) <= \*(vec + i + 1)){

flag = 0;

break;

}

if(flag){

printf("Vector is descending\n");

return 0;

}

int MinId = 0;

for(int i = 1; i < VecLength; i++)

if (vec[i] < vec[MinId])

MinId = i;

float tmp;

tmp = vec[VecLength - 1];

vec[VecLength - 1] = vec[MinId];

vec[MinId] = tmp;

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

printf("%0.3f ", \*(vec + i));

}

printf("\n");

return 0;

}



**Додаток 2.**

Назва файлу: **cycleshift.c**

#include <stdio.h>

int main(){

const int Shift = 2;

int matrix[128][128];

int n, m;

printf("Enter height and width of your matrix: ");

scanf("%d %d", &n, &m);

if (n == 0 || m == 0){

printf("Height or width should be at least 1!\n");

return 1;

}

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

printf("Enter the row #%d: ", i + 1);

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

scanf("%d", \*(matrix + i) + j);

}

}

printf("Result of 2-up cyclic shift: ");

if (n == 1){

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

printf("%d ", matrix[i][0]);

printf("\n");

return 0;

}

int res[n][m];

for(int i = Shift; i < n; ++i){

for(int j = 0; j < m; ++j)

res[i - Shift][j] = matrix[i][j];

}

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

for(int j = 0; j < m; ++j)

\*(\*(res + i + n - Shift) + j) = \*(\*(matrix + i) + j);

}

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

printf("\n");

for(int j = 0; j < m; ++j)

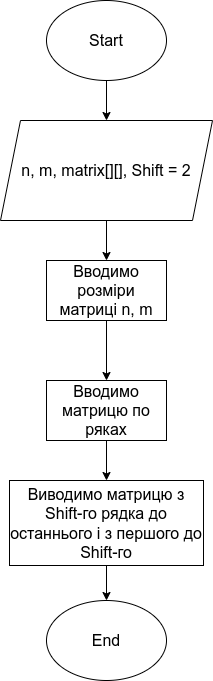
printf("%d ", res[i][j]);

}

printf("\n");

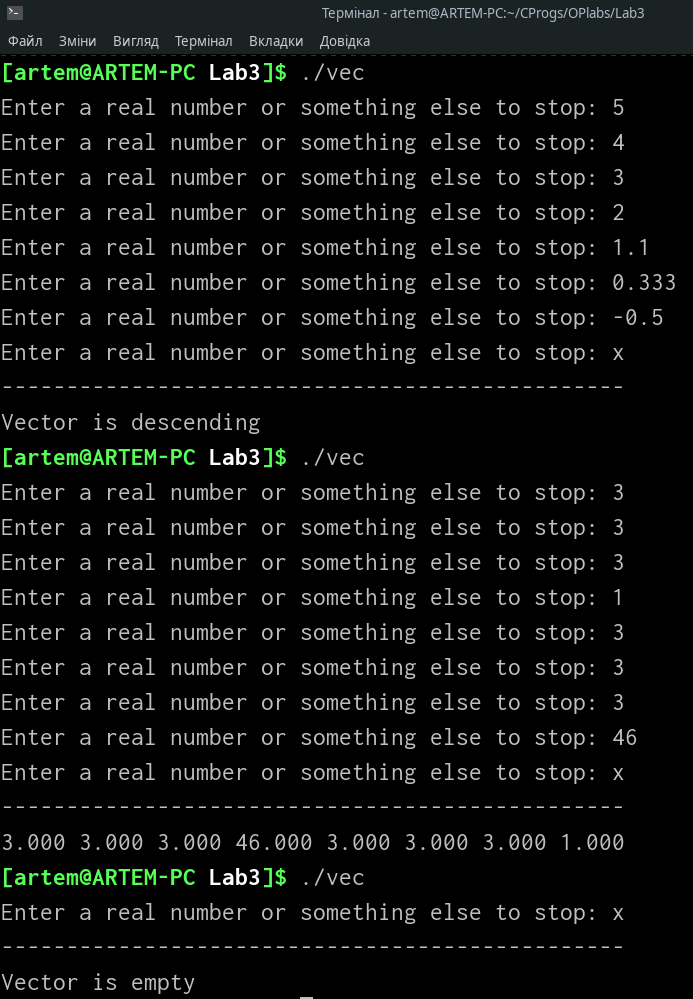
return 0;

}

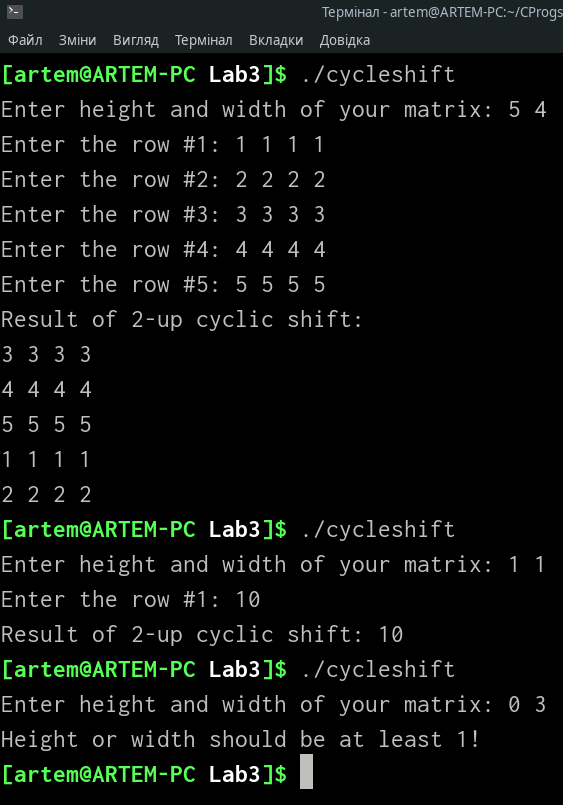


**Протокол роботи**

**Додаток 1: vec.c**



**Додаток 2: cycleshift.c :**



**Висновки**

Масиви дають змогу зберігати однотипні дані, полегшуючи їх обробку.