Міністерство освіти і науки України

Дніпропетровський національний університет

імені Олеся Гончара

ФФЕКС

Кафедра ЕОМ

**Звіз з лабораторної роботи № 1**

Виконав: студент 2 курсу

Шейко Р.О.

Науковий керівник:

Викладач кафедри EOM

Спірінцева Ольга Володимирівна

м. Дніпро

2022

Завдання 1. Варіант 3

Частина 1.  
Одновимірний символьний масив;

Покажчик на тип char;

Статичний одновимірний масив довгих чисел;

Покажчик на масив довгих цілих чисел;

Тривимірний масив плаваючих чисел;

Покажчик на двовимірний масив плаваючих чисел.

Ілюстрації роботи коду у консолі наведені у рисунках нижче.

*Лістинг 1.*

setlocale(LC\_ALL, "Rus"); //Язык

char massive[6] = {'9', '9' ,'9' ,'9' ,'9' };

char\* pointer\_massive;

static long matrix[] = {10, 234};

long\* pointer\_matrix;

float matrix\_3[2][3][2];

float\*\*\* matrix\_3point;

Завдання 2.

2.1. Перевірити вміст масиву №1 (за допомогою циклу for та операції виведення cout<<).

*Лістинг 2.*

puts("\n2.1");

for (int i = 0; i < strlen(massive); i++)

printf("%d. %c\n", i, massive[i]);



Рис 1.

2.2. Ввести дані масив №1 (за допомогою циклу for і операції введення cin>>).

*Лістинг 3.*

puts("\n2.2");

puts("\nЗаполните массив данными: ");

for (int i = 0; i < strlen(massive); i++)

scanf\_s("%c", &massive[i]);

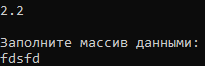


Рис 2.

2.3. Ще раз перевірити вміст цього масиву, зробити висновки.

*Лістинг 4.*

puts("\n2.3");

for (int i = 0; i < strlen(massive); i++)

printf("%d. %c\n", i, massive[i]);



Рис 3.

Висновок; Я можу змінювати елементу масиву з тим типом даних, що йому призначено.

2.4 Присвоїти вказівнику №2 адресу масиву №1, вивести на екран адреси масиву та вказівника та вміст покажчика. Зробити висновки.

*Лістинг 5.*

puts("\n2.4");

pointer\_matrix = massive;

printf("Адресс указателя - %p\n", pointer\_matrix);

printf("Адресс массива - %p\n", massive);

printf("Содержимое указателя - %s\n", pointer\_matrix);

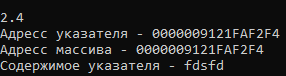


Рис 4.

Висновок; Присвоювання вказівників можливо у будь-якому варіанті. Масив є pointer`ом. Вказівнику можливо присвоїти вказівник початку масиву. Також можливий й output цїєї інформації в С, хоча типи даних різні…

2.5. Повторити пункт 3 для покажчика, що містить адресу масиву. Зробити висновки.

*Лістинг 6.*

puts("\n2.5");

pointer\_massive = massive;

for (int i = 0; i < strlen(massive); i++)

printf("%d. %c\n", i, \*(pointer\_massive + i));



Рис 5.

Висновок; так як ми присвоїли адресу масиву(тобто поставили вказівник на перший елемент масиву), то і out put буде таким же.

2.6. Повторити пункти 1 – 3 для статичного масиву №3. Зробити висновки.

*Лістинг 7.*

puts("\n2.6");

for (int i = 0; i < sizeof(matrix) / sizeof(matrix[0]); i++)

printf("%d. %i\n", i, matrix[i]);

puts("\nЗаполните массив данными(вводите два числа, третье дял подтверждения): ");

for (int i = 0; i < sizeof(matrix) / sizeof(matrix[0]); i++)

scanf\_s("%i\n", &matrix[i]);

for (int i = 0; i < sizeof(matrix) / sizeof(matrix[0]); i++)

printf("%d. %i\n", i, matrix[i]);

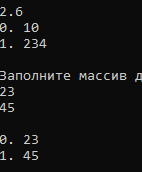


Рис 6.

Висновок. Статичний масив змінюється. Його поведінка не відрізняється від default massive.

2.7. Використовуючи наявний покажчик №2, створити динамічний масив та повторити для нього пункти 1 – 3. Зробити висновки.

*Лістинг 8.*

puts("\n2.7");

int size = 2;

long \*pointer\_matrix\_ = (long\*)malloc(size \* sizeof(long));

long matrix\_m[2];

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

pointer\_matrix\_[i] = 0;

for (int i = 0; i < sizeof(pointer\_matrix\_) / sizeof(long); i++)

printf("%d. %i\n", i, pointer\_matrix\_[i]);

puts("\nЗаполните массив данными(вводите два числа, третье дял подтверждения): ");

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

{

long num = 0;

scanf\_s("%ld\n", &num);

\*(pointer\_matrix\_ + i) = num;

matrix\_m[i] = num;

}

puts("\nУказатель массив: ");

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

printf("%i. %ld\n", i, \*(pointer\_matrix + i));

puts("\nМассив обычный: ");

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

printf("%i. %ld\n", i, matrix\_m[i]);

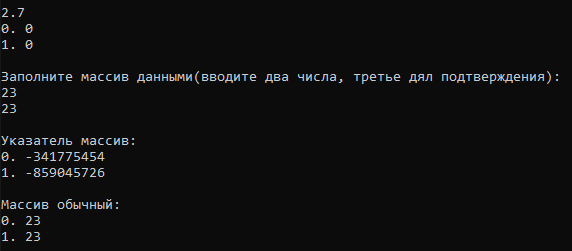


Рис 7.

Висновок. Записувати числа через вказівник неможливо. Запис до стандартного масиву можлива. У промисловості використовуй масиви, бо найважливіше – це читаемість коду для інших співробітників, а не боротьба за 4 байту пам’яті.

2.8. Видалити динамічний масив. Використовуючи покажчик №4, створити двовимірний динамічний масив та повторити для нього пункти 2, 3. Зробити висновки. Видалити двовимірний динамічний масив.

*Лістинг 9.*

*//доп functions*

void client\_input(int\*\* massive, int N) {

char temp[K];

int pomilka = 0;

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

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

{

do {

pomilka = 0;

printf("Мatrix[%d][%d] = ", i, j);

scanf\_s("%s", &temp, K - 1);

for (int a = 0; a < strlen(temp); a++)

if (atoi(temp) == 0 && !(('0' <= temp[a]) && (temp[a] <= '9')))

pomilka = 1;

if (pomilka == 1)

printf("Вводите только числа!\n");

else {

massive[i][j] = atoi(temp);

break;

}

} while (1);

}

}

void out\_matrix(int\*\* massive, int N) {

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

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

printf("%4d", massive[i][j]);

puts("");

}

}

*//main function*

puts("\n2.8");

free(pointer\_matrix\_);

int N = 2 ;

int\*\* matrix\_2d = (int\*\*)malloc(N \* sizeof(int\*));

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

matrix\_2d[i] = (int\*)malloc(N \* sizeof(int));

client\_input(matrix\_2d, N);

out\_matrix(matrix\_2d, N);

free(matrix\_2d);

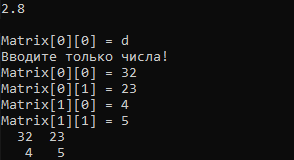


Рис 8.

Висновок: Можна створювати динамічні двовимірні масиви. Вони працюють)

2.9 Вивести на екран будь-який із елементів тривимірного масиву №5, використовуючи операцію індексації.

*Лістинг 10.*

puts("\n2.9");

for (int i = 0; i < 2; i++)

for (int j = 0; j < 3; j++)

for (int k = 0; k < 2; k++)

matrix\_3[i][j][k] = (float)i + j + k;

printf("Елемент трехмерного массива по индексу: %f", matrix\_3[1][2][1]);



Рис 9.

2.10 Повторити пункт 9, використовуючи ім'я масиву як вказівник та операцію доступу за вказівником.

*Лістинг 11.*

puts("\n2.10");

printf("Елемент трехмерного массива по указателю: %f", \*(\* (\*(matrix\_3 + 1) + 2) + 1) );



Рис 10

2.11 Присвоїти покажчику №6 на двовимірний масив адресу тривимірного масиву №5. Повторити для цього покажчика пункт 10. Зробити висновки.

*Лістинг 12.*

puts("\n2.11");

matrix\_3point = &matrix\_3;

printf("Елемент трехмерного массива по указателю через указатель: %f", \*(matrix\_3point + 1 \* 3 + 2 \* 2 + 1));

puts("");



Рис 11

Висновок загалом.

Мова С є дуже нестандартною мовою програмування. Робота з масивами у неї дуже сильно ускладнена, зокрема вказівниками. Вказівники є великою “каменякою” у огороді мови С та С++, за їх небезпечністю взагалом для комерції.

Робота з масивами є у всіх мовах програмування. Від неї не можна відійти, але, наприклад, у мові програмування С# є так званий List<>. Він дуже сильно спрощує розробку, роботу та читання коду. Той же самий Queue<> та інші варіації праці зі структурами даних або класами є дуже вигідним у відносно час/швидкості роботи. Тому я вважаю, що на другому курсі масиви і робота з ними є відпрацьована. Вона не потрібна у повторенні.

Лістинг з лабораторної роботи. Файл <main.c>

#include <stdio.h>

#include <locale.h>

#include <stdlib.h>

#include <string.h>

#include "AiSD\_L1.h"

#define K 50

void client\_input(int\*\* massive, int N) {

char temp[K];

int pomilka = 0;

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

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

{

do {

pomilka = 0;

printf("Мatrix[%d][%d] = ", i, j);

scanf\_s("%s", &temp, K - 1);

for (int a = 0; a < strlen(temp); a++)

if (atoi(temp) == 0 && !(('0' <= temp[a]) && (temp[a] <= '9')))

pomilka = 1;

if (pomilka == 1)

printf("Вводите только числа!\n");

else {

massive[i][j] = atoi(temp);

break;

}

} while (1);

}

}

void out\_matrix(int\*\* massive, int N) {

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

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

printf("%4d", massive[i][j]);

puts("");

}

}

int main()

{

setlocale(LC\_ALL, "Rus"); //Язык

char massive[6] = {'9', '9' ,'9' ,'9' ,'9' };

char\* pointer\_massive;

static long matrix[] = {10, 234};

long\* pointer\_matrix;

//2.1

puts("\n2.1");

for (int i = 0; i < strlen(massive); i++)

printf("%d. %c\n", i, massive[i]);

//2.2

puts("\n2.2");

puts("\nЗаполните массив данными: ");

for (int i = 0; i < strlen(massive); i++)

scanf\_s("%c", &massive[i]);

//2.3

puts("\n2.3");

for (int i = 0; i < strlen(massive); i++)

printf("%d. %c\n", i, massive[i]);

//2.4

puts("\n2.4");

pointer\_matrix = massive;

printf("Адресс указателя - %p\n", pointer\_matrix);

printf("Адресс массива - %p\n", massive);

printf("Содержимое указателя - %s\n", pointer\_matrix);

//2.5

puts("\n2.5");

pointer\_massive = massive;

for (int i = 0; i < strlen(massive); i++)

printf("%d. %c\n", i, \*(pointer\_massive + i));

//2.6

puts("\n2.6");

for (int i = 0; i < sizeof(matrix) / sizeof(matrix[0]); i++)

printf("%d. %i\n", i, matrix[i]);

puts("\nЗаполните массив данными(вводите два числа, третье дял подтверждения): ");

for (int i = 0; i < sizeof(matrix) / sizeof(matrix[0]); i++)

scanf\_s("%i\n", &matrix[i]);

for (int i = 0; i < sizeof(matrix) / sizeof(matrix[0]); i++)

printf("%d. %i\n", i, matrix[i]);

//2.7

puts("\n2.7");

int size = 2;

long \*pointer\_matrix\_ = (long\*)malloc(size \* sizeof(long));

long matrix\_m[2];

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

pointer\_matrix\_[i] = 0;

for (int i = 0; i < sizeof(pointer\_matrix\_) / sizeof(long); i++)

printf("%d. %i\n", i, pointer\_matrix\_[i]);

puts("\nЗаполните массив данными(вводите два числа, третье дял подтверждения): ");

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

{

long num = 0;

scanf\_s("%ld\n", &num);

\*(pointer\_matrix\_ + i) = num;

matrix\_m[i] = num;

}

puts("\nУказатель массив: ");

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

printf("%i. %ld\n", i, \*(pointer\_matrix + i));

puts("\nМассив обычный: ");

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

printf("%i. %ld\n", i, matrix\_m[i]);

//2.8

puts("\n2.8");

free(pointer\_matrix\_);

int N = 2

;

int\*\* matrix\_2d = (int\*\*)malloc(N \* sizeof(int\*));

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

matrix\_2d[i] = (int\*)malloc(N \* sizeof(int));

client\_input(matrix\_2d, N);

out\_matrix(matrix\_2d, N);

free(matrix\_2d);

//2.9

puts("\n2.9");

float matrix\_3[2][3][2];

for (int i = 0; i < 2; i++)

for (int j = 0; j < 3; j++)

for (int k = 0; k < 2; k++)

matrix\_3[i][j][k] = (float)i + j + k;

printf("Елемент трехмерного массива по индексу: %f", matrix\_3[1][2][1]);

//2.10

puts("\n2.10");

printf("Елемент трехмерного массива по указателю: %f", \*(\* (\*(matrix\_3 + 1) + 2) + 1) );

//2.11

puts("\n2.11");

float\*\*\* matrix\_3point = &matrix\_3;

printf("Елемент трехмерного массива по указателю через указатель: %f", \*(matrix\_3point + 1 \* 3 + 2 \* 2 + 1));

puts("");

}