МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпропетровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №7**

**з дисципліни «Основи програмування»**

**на тему: «Функції. Методи передачі параметрів. Багатофайлові програми.»**

Виконав: студент гр. ПЗ1911

Сіньков Г.О.

Прийняла: ас. каф. КІТ

Нежуміра О. І.

Дніпро, 2019

**Лабораторна робота №7**

**Тема.** Функції. Методи передачі параметрів. Багатофайлові програми.

**Мета.**

1) Вивчити організацію функцій і механізм передачі параметрів. Отримати практичні навички розробки та використання функцій.

2) Вивчити призначення і організацію багатофайлових програми. Отримати практичні навички розробки багатофайлових програм.

**Постановка задачі №1**

1.Загальна постановка: розробити програму для демонстрації механізму передачі параметрів. Для вказаних способів передачі параметрів і типів параметрів написати функції, в яких параметри змінюють своє значення (можна використати арифметичні операції, присвоювання тощо). Функції повинні повертати значення заданого типу.

Для виклику кожної функції передбачити виведення на консоль:

1) назви способу передачі параметрів у функцію;

2) значень параметрів, що передаються у функцію, до і після її виклику;

3) значення, яке повернула функція.

2.Індивідуальна постановка:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Типи параметрів | Спосіб передачі параметрів | Тип значення, що повертає функцію | Рівень скількість | | | | |
| E | D | C | B | A |
| int, char, bool | за значенням | int | + | + | + | + | + |
| int, char, bool | за посиланням | char | + | + | + | + | + |
| int, char, bool | за вказівником | void | + | + | + | + | + |
| int, char, bool | за значенням | int\* |  | + | + | + | + |
| int, char, bool | за посиланням | char\* |  | + | + | + | + |
| числовий масив | за значенням | void |  |  | + | + | + |
| числовий масив | за посиланням | void |  |  | + | + | + |
| числовий масив | за вказівником | void |  |  | + | + | + |
| структура | за значенням | структура |  |  |  | + | + |
| структура | за посиланням | вказівник на структуру |  |  |  | + | + |
| структура | за вказівником | вказівник на структуру |  |  |  | + | + |
| символьний масив | за значенням | void |  |  |  |  | + |
| символьний масив | за посиланням | void |  |  |  |  | + |
| символьний масив | за вказівником | void |  |  |  |  | + |

**Текст програми**

metod.h

struct struct\_type

{

int a, b;

char ch3;

};

int f1(int x, char ch, bool f);

int f2(int &x, char &ch, bool &f);

void f3(int\*, char \*ch, bool \*f);

int\* f4(int x, char ch, bool f);

char\* f5(int &x, char &ch, bool &f);

void f6(int arr[], int size);

void f7(int &arr, int size);

void f8(int \*arr, int size);

void f9(struct struct\_type parm);

void f10(struct\_type \*arg);

void f11(struct\_type &arg);

metod.cpp

#include <iostream>

#include "metod.h"

int f1(int x, char ch, bool f)

{

std::cout << "За значеню" << std::endl;

x += x;

ch = 'a';

f = !f;

return x;

}

int f2(int &x, char &ch, bool &f)

{

std::cout << "За посиланням" << std::endl;

x += x;

ch = 'b';

f = !f;

return ch;

}

void f3(int \*x, char \*ch, bool \*f)

{

std::cout << "За вказівником" << std::endl;

\*x += \*x;

\*ch = 'c';

\*f = !(\*f);

}

int\* f4(int x, char ch, bool f)

{

std::cout << "За посиланням" << std::endl;

x += x;

ch = 'd';

f = !f;

return &x;

}

char\* f5(int &x, char &ch, bool &f)

{

std::cout << "За посиланням" << std::endl;

x += x;

ch = 'e';

f = !f;

return &ch;

}

void f6(int arr[], int size)

{

int sum = 0;

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

{

arr[i] = rand() % 10;

}

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

{

sum += arr[i];

}

std::cout << "Сума: " << sum << std::endl;

std::cout << "Функція випадково перезаповняє масив, також рахує суму елементів. Не повертає значення" << std::endl;

}

void f7(int &arr, int size)

{

int sum = 0;

int \*p = &arr;

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

{

p[i] = rand() % 10;

}

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

{

sum += p[i];

}

std::cout << "Сума: " << sum << std::endl;

std::cout << "Функція випадково перезаповняє масив, також рахує суму елементів. Не повертає значення" << std::endl;

}

void f8(int \*arr, int size)

{

int sum = 0;

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

{

arr[i] = rand() % 10;

}

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

{

sum += arr[i];

}

std::cout << "Сума: " << sum << std::endl;

std::cout << "Функція випадково перезаповняє масив, також рахує суму елементів. Не повертає значення" << std::endl;

}

void f9(struct struct\_type parm)

{

parm.a = 200;

parm.b = 40;

parm.ch3 = 'f';

std::cout << "Всі зпначення структури: " << parm.a << ", " << parm.b << ", " << parm.ch3 << std::endl;

}

void f10(struct\_type \*arg)

{

arg->a = 200;

arg->b = 40;

arg->ch3 = 'f';

std::cout << "Всі зпначення структури: " << arg->a << ", " << arg->b << ", " << arg->ch3 << std::endl;

}

void f11(struct\_type &arg)

{

arg.a = 200;

arg.b = 40;

arg.ch3 = 'f';

std::cout << "Всі зпначення структури: " << arg.a << ", " << arg.b << ", " << arg.ch3 << std::endl;

}

implementation.h

#include <iostream>

#include "Windows.h"

#include "metod.h"

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

///////////////////////////////////////

int x = 10;

char ch1 = (43);

bool question = true;

std::cout << std::endl;

std::cout << "До виклика: " << x << '\t' << ch1 << '\t' << question << std::endl;

int n = f1(10, 43, question);

std::cout << "Функція 1 повертає значення int: " << n << std::endl;

std::cout << "Після виклика: " << x << '\t' << ch1 << '\t' << question << std::endl << std::endl;

std::cout << std::endl;

std::cout << "До виклика: " << x << '\t' << ch1 << '\t' << question << std::endl;

char ch = f2(x, ch1, question);

std::cout << "Функція 2 повертає значення char: " << ch << std::endl;

std::cout << "Після виклика: " << x << '\t' << ch1 << '\t' << question << std::endl << std::endl;

std::cout << std::endl;

std::cout << "До виклика: " << x << '\t' << ch1 << '\t' << question << std::endl;

f3(&x, &ch1, &question);

std::cout << "Не повертає значення" << std::endl;

std::cout << "Після виклика: " << x << '\t' << ch1 << '\t' << question << std::endl << std::endl;

///////////////////////////////////////

std::cout << std::endl;

std::cout << "До виклика: " << x << '\t' << ch1 << '\t' << question << std::endl;

int \*m = f4(x, ch1, question);

std::cout << "Функція 1 повертає значення int: " << n << std::endl;

std::cout << "Після виклика: " << x << '\t' << ch1 << '\t' << question << std::endl << std::endl;

std::cout << std::endl;

std::cout << "До виклика: " << x << '\t' << ch1 << '\t' << question << std::endl;

char \*p = f5(x, ch1, question);

std::cout << "Функція 5 повертає значення int: " << n << std::endl;

std::cout << "Після виклика: " << x << '\t' << ch1 << '\t' << question << std::endl << std::endl;

///////////////////////////////////////

int arr[] = { 1, 2, 3 };

std::cout << "До виклика: ";

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

{

std::cout << arr[i];

}

std::cout << std::endl;

f6(arr, sizeof(arr) / sizeof(arr[0]));

std::cout << "Після виклика: ";

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

{

std::cout << arr[i];

}

std::cout << std::endl;

std::cout << std::endl;

std::cout << "До виклика: ";

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

{

std::cout << arr[i];

}

std::cout << std::endl;

f7(\*arr, sizeof(arr) / sizeof(arr[0]));

std::cout << "Після виклика: ";

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

{

std::cout << arr[i];

}

std::cout << std::endl;

std::cout << std::endl;

std::cout << "До виклика: ";

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

{

std::cout << arr[i];

}

std::cout << std::endl;

f8(arr, sizeof(arr) / sizeof(arr[0]));

std::cout << "Після виклика: ";

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

{

std::cout << arr[i];

}

std::cout << std::endl;

///////////////////////////////////////

struct\_type arg;

arg = { 100, 20, 'g' };

//struct arg \*p = &arg;

std::cout << std::endl;

std::cout << "Всі значення структури до виклика структури: " << arg.a << ", " << arg.b << ", " << arg.ch3 << std::endl;

f9(arg);

std::cout << "Всі значення структури після виклика структури: " << arg.a << ", " << arg.b << ", " << arg.ch3 << std::endl;

std::cout << std::endl;

std::cout << std::endl;

std::cout << "Всі значення структури до виклика структури: " << arg.a << ", " << arg.b << ", " << arg.ch3 << std::endl;

f10(&arg);

std::cout << "Всі значення структури після виклика структури: " << arg.a << ", " << arg.b << ", " << arg.ch3 << std::endl;

std::cout << std::endl;

std::cout << std::endl;

std::cout << "Всі значення структури до виклика структури: " << arg.a << ", " << arg.b << ", " << arg.ch3 << std::endl;

f11(arg);

std::cout << "Всі значення структури після виклика структури: " << arg.a << ", " << arg.b << ", " << arg.ch3 << std::endl;

std::cout << std::endl;

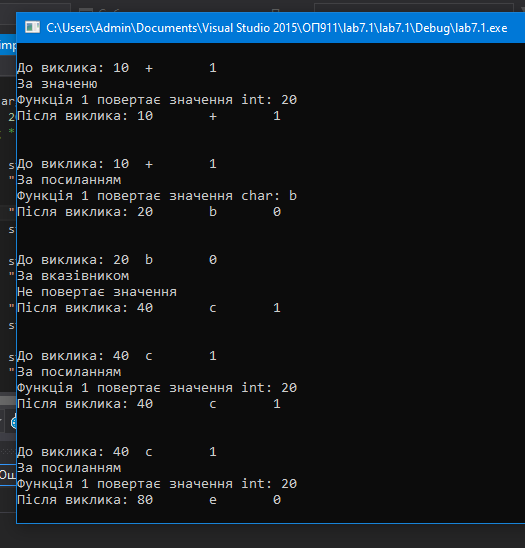
system("pause");

return 0;

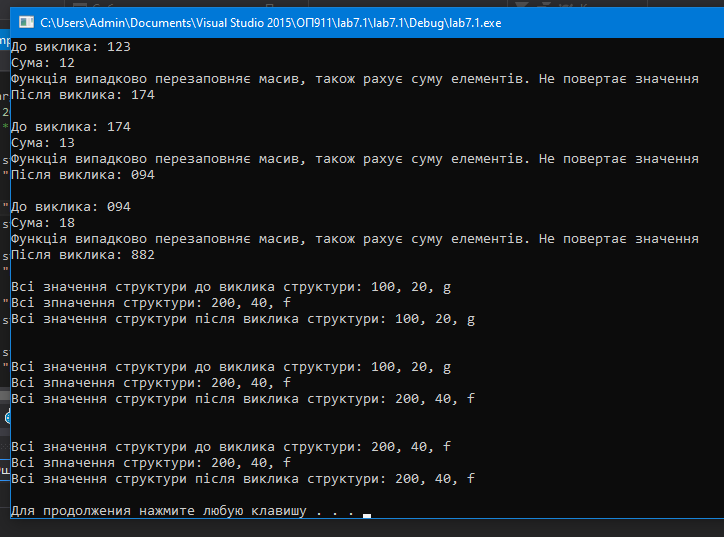
}

**Результати виповнення програми**

1)



2)



Висновок: теоретично у програмуванні розрізняють шість способів передачі фактичних параметрів у підпрограму, у OOo Basic (Visual Basic, VBA) реалізовано два з них: за значенням (ByVal) та за адресою (ByRef).

Під час передачі за значенням у підпрограму передається лише значення фактичного параметра, відповідно, зміни такого параметра всередині підпрограми не помітні у основній програмі. Фактичним параметром, що передається за значенням, може бути як змінна, так і рядкова або іменована константа, або навіть деякий вираз, результат обчислення якого передається в підпрограму.

Під час передачі за адресою (посиланням) у підпрограму передається не саме значення а фізична адреса змінної — фактичного параметра. У цьому випадку підпрограма працює із переданою змінною як із власною, і результат цієї роботи можна побачити у основній програмі. Фактичним парметром, переданим за адресою може бути лише змінна, якщо параметрам описаним з ключовим словом ByRef передати константу або значення виразу (які не мають фізичної адреси у пам’яті ПК), то Basic буде працювати з ними, як із параметрами-значеннями.

**Постановка задачі №2**

Переробити завдання з лабораторної роботи №5 в багатофайлову програму. Для кожної задачі розробити функцію (її специфікацію записати в коді програми). Розбити програму з трьох файлів: головний файл, файл заголовків і файл реалізації (або більше за функціоналом програми). В головному файлі реалізувати меню користувача. В файл заголовків перенести константи, опис типів даних, прототипи функцій. У файлі реалізації розмістити визначення функцій.

**Текст програми**

metod.h

struct address {

char street[50];

int building;

int number;

};

struct apartment {

char name[50];

address built;

int price;

};

void input\_apa(int& quantity, apartment \*house);

void output\_apa(int& quantity, apartment \*house);

void search\_by\_price(int& quantity, apartment \*house);

void minimum\_price(int& quantity, apartment \*house);

void coincides\_apa(int& quantity, apartment \*house);

metod.cpp

//Програма з власним меню для роботи зі структурними типами даних

/\*Програма повинна додавати реквізити квартир в список структур, виводити список руквізитів квартири на екран,

виведення списку квартир вказаної ціни, пошук квартир з мінімальною ціною, пошук вказаної квартири в списку\*/

#include <iostream>

#include "metod.h"

void input\_apa(int& quantity, apartment \*house) // додавання квартир в список

{

std::system("cls");

std::cout << "Номер квартири у списку: " << quantity + 1 << std::endl;

std::cout << "Введіть ФІО: ";

std::cin.getline(house[quantity].name, 50);

std::cout << "Введіть адрес: ";

std::cin.getline(house[quantity].built.street, 50);

std::cout << "Введіть номер будинка: ";

std::cin >> house[quantity].built.building;

std::cout << "Введіть номер квартири: ";

std::cin >> house[quantity].built.number;

std::cout << "Введіть ціну квартири: ";

std::cin >> house[quantity].price;

quantity++;

}

void output\_apa(int& quantity, apartment \*house) // вивід списку всіх квартир

{

std::system("cls");

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

{

std::cout << "№:" << i + 1 << std::endl;

std::cout << " ФІО: " << house[i].name << std::endl;

std::cout << " Адрес: " << house[i].built.street << ", " << house[i].built.building << ", " << house[i].built.number << std::endl;

std::cout << " Ціна: " << house[i].price << "$" << std::endl << std::endl;

}

system("pause");

}

void search\_by\_price(int& quantity, apartment \*house) // Виведення списку квартир вказаної ціни

{

std::system("cls");

int search\_price;

bool find = false;

std::cout << "Введіть ціну квартири: ";

std::cin >> search\_price;

std::cin.get(); // очистка буфера

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

{

if (search\_price == house[i].price)

{

std::cout << "№:" << i + 1 << std::endl;

std::cout << " ФІО: " << house[i].name << std::endl;

std::cout << " Адрес: " << house[i].built.street << ", " << house[i].built.building << ", " << house[i].built.number << std::endl;

std::cout << " Ціна: " << house[i].price << "$" << std::endl << std::endl;

find = true;

system("pause");

}

}

if (!find)

{

std::cout << "Не знайдено в списку кватири з такою ціною!" << std::endl;

std::cout << std::endl;

system("pause");

}

}

void minimum\_price(int& quantity, apartment \*house) // Пошук квартир з мінімальною ціною

{

std::system("cls");

int min\_price = house[0].price;

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

{

if (min\_price < house[i].price)

{

//std::cout << "Найменьша ціна " << min\_price << std::endl;

}

else

{

min\_price = house[i].price;

}

}

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

{

if (min\_price == house[i].price)

{

std::cout << "№:" << i + 1 << std::endl;

std::cout << " ФІО: " << house[i].name << std::endl;

std::cout << " Адрес: " << house[i].built.street << ", " << house[i].built.building << ", " << house[i].built.number << std::endl;

std::cout << " Ціна: " << house[i].price << "$" << std::endl << std::endl;

}

}

system("pause");

}

void coincides\_apa(int& quantity, apartment \*house) // Пошук вказаної квартири в списку

{

std::system("cls");

char name\_[50];

std::cout << "Введіть ФІО: ";

std::cin.getline(name\_, 50);

char street\_[50];

std::cout << "Введіть адрес: ";

std::cin.getline(street\_, 50);

int building\_;

std::cout << "Введіть номер будинка: ";

std::cin >> building\_;

int number\_;

std::cout << "Введіть номер квартири: ";

std::cin >> number\_;

int price\_;

std::cout << "Введіть ціну квартири: ";

std::cin >> price\_;

bool find = false;

for (int i = 0; !(find) && i < quantity; i++)

{

if (strcmp(name\_, house[i].name) == 0)

{

if (strcmp(street\_, house[i].built.street) == 0)

{

if (building\_ == house[i].built.building)

{

if (number\_ == house[i].built.number)

{

if (price\_ == house[i].price)

{

std::cout << std::endl << "Квартира с такими реквізитами є у списку." << std::endl;

std::cout << std::endl;

find = true;

system("pause");

}

else

{

std::cout << "Не має квартири в списку. Можливо ви помилилися з ціню!" << std::endl;

find = true;

system("pause");

}

}

else

{

std::cout << "Не має квартири в списку. Можливо ви помилилися з номером квартири!" << std::endl;

find = true;

system("pause");

}

}

else

{

std::cout << "Не має квартири в списку. Можливо ви помилилися з номером будинка!" << std::endl;

find = true;

system("pause");

}

}

else

{

std::cout << "Не має квартири в списку. Можливо ви помилилися з вулицею!" << std::endl;

find = true;

system("pause");

}

}

else

{

std::cout << "Не має квартири в списку. Можливо ви помилилися з ініціалами!" << std::endl;

find = true;

system("pause");

}

}

}

source.cpp

//Програма з власним меню для роботи зі структурними типами даних

/\*Програма повинна додавати реквізити квартир в список структур, виводити список руквізитів квартири на екран,

виведення списку квартир вказаної ціни, пошук квартир з мінімальною ціною, пошук вказаної квартири в списку\*/

#include <iostream>

#include <Windows.h>

#include <string>

#include <cstdlib>

#include "metod.h"

void main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int quantity = 2; // кількість введених квартир в структуру

apartment house[10] = { { "Сідоров Заур Данилович", "Карла Маркса", 555, 55, 55000 },

{ "Семёнов Родион Петрович", "Кіров", 485, 6, 45000 }

};// количество уже введеных номеров

int menu\_paragraph;

do

{

std::system("cls");

std::cout << "1. Додавання квартир у список\n";//E ч.1 +

std::cout << "2. Виведення квартир у список\n";//E ч.2 +

std::cout << "3. Виведення списку квартир вказаної ціни\n";//D +

std::cout << "4. Пошук квартир з мінімальною ціною\n";//C +

std::cout << "5. Пошук вказаної квартири в списку\n";//B +

std::cout << "6. Вихід\n";

while (!(std::cin >> menu\_paragraph) || (std::cin.peek() != '\n')) // цикл для перевірки введеної зміної на символи

{

std::system("cls");

std::cout << "1. Додавання квартир у список\n";

std::cout << "2. Виведення квартир у список\n";

std::cout << "3. Виведення списку квартир вказаної ціни\n";

std::cout << "4. Пошук квартир з мінімальною ціною\n";

std::cout << "5. Пошук вказаної квартири в списку\n";

std::cout << "6. Вихід\n";

std::cin.clear();

while (std::cin.get() != '\n');

std::cerr << "Помилка вводу! Ввод номера меню повино бути цілим" << std::endl;

std::cout << std::endl;

}

std::cin.get();//очистка буфера

switch (menu\_paragraph)

{

case 1:

input\_apa(quantity, house);

break;

case 2:

output\_apa(quantity, house);

break;

case 3:

search\_by\_price(quantity, house);

break;

case 4:

minimum\_price(quantity, house);

break;

case 5:

coincides\_apa(quantity, house);

break;

case 6:

return;

break;

default:

break;

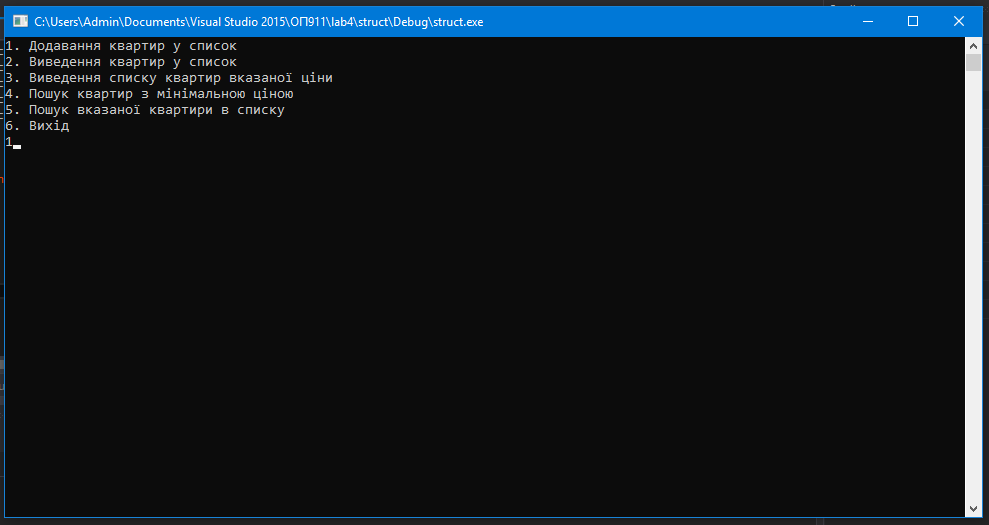
}

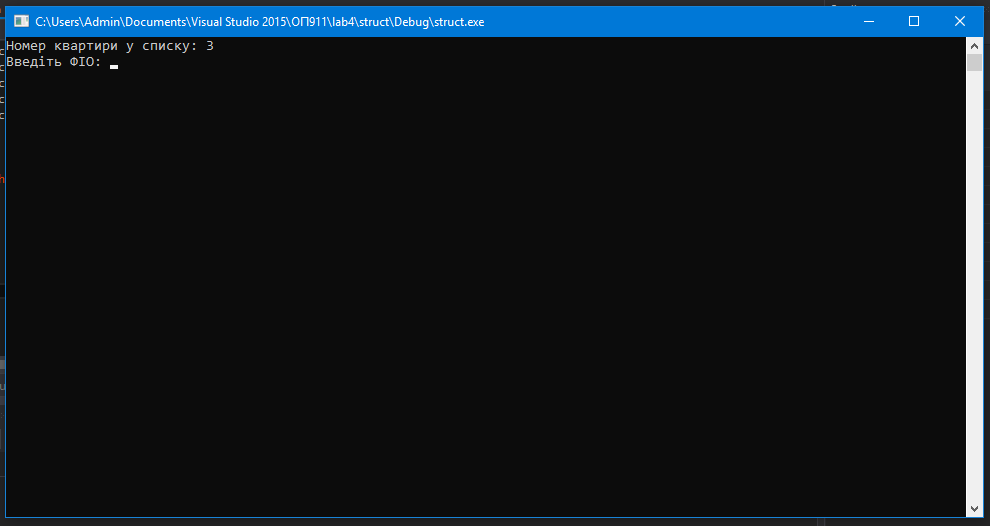
} while (menu\_paragraph != 6);

}

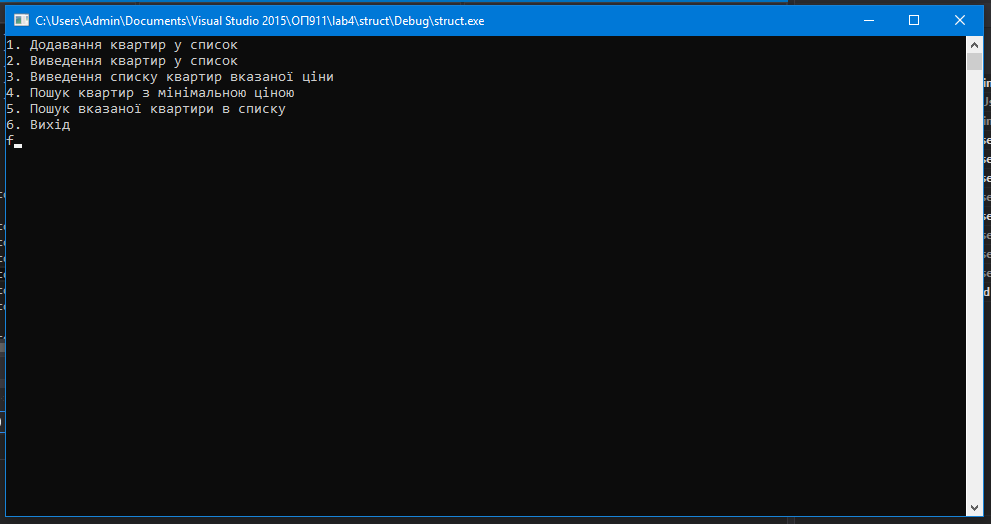
**Результати виповнення програми**

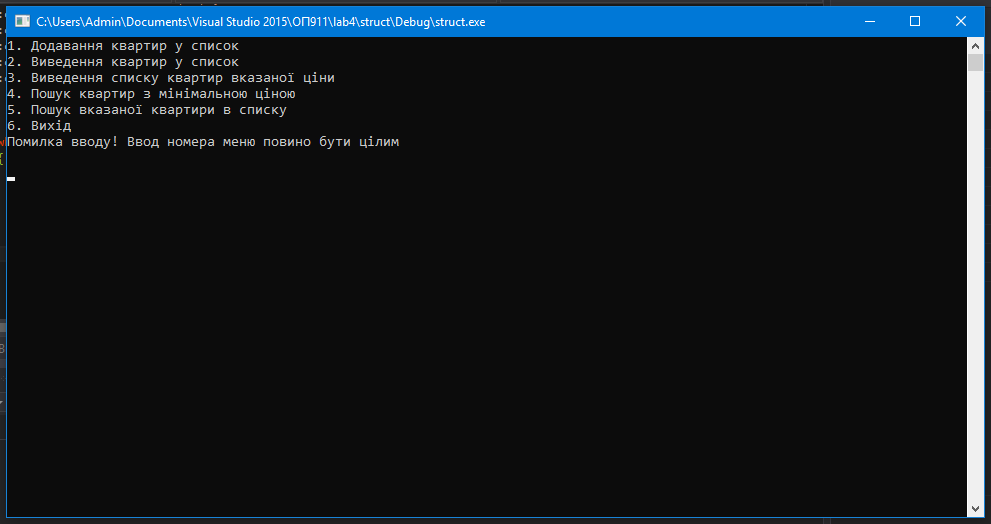
1)



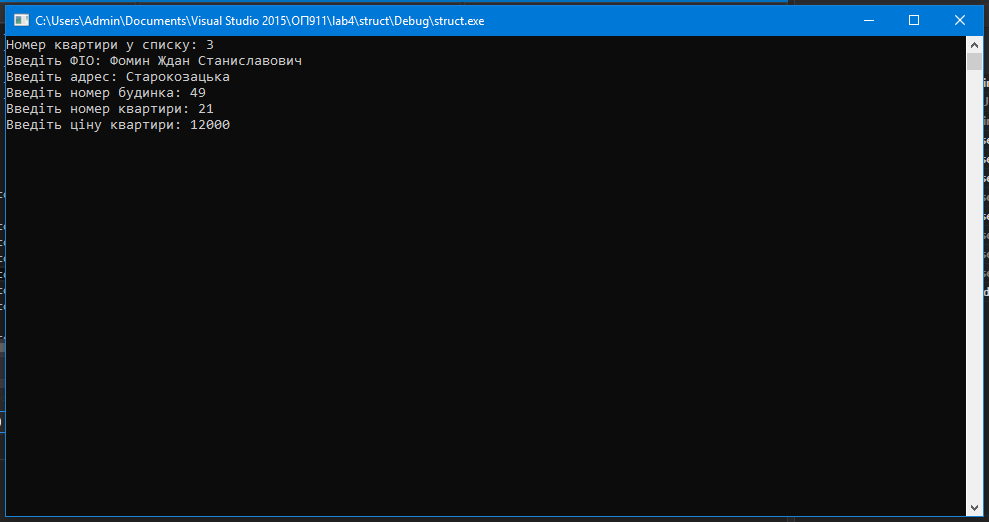


2)

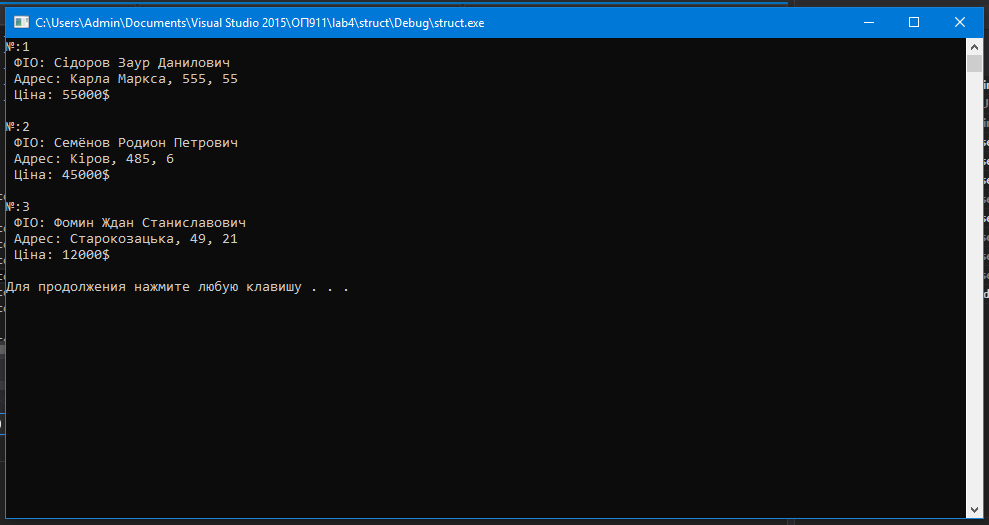




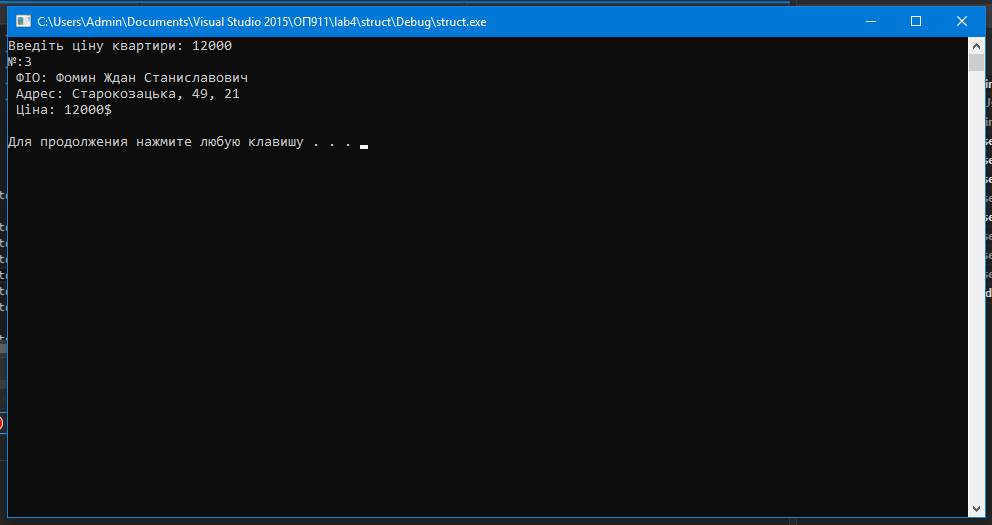
3)



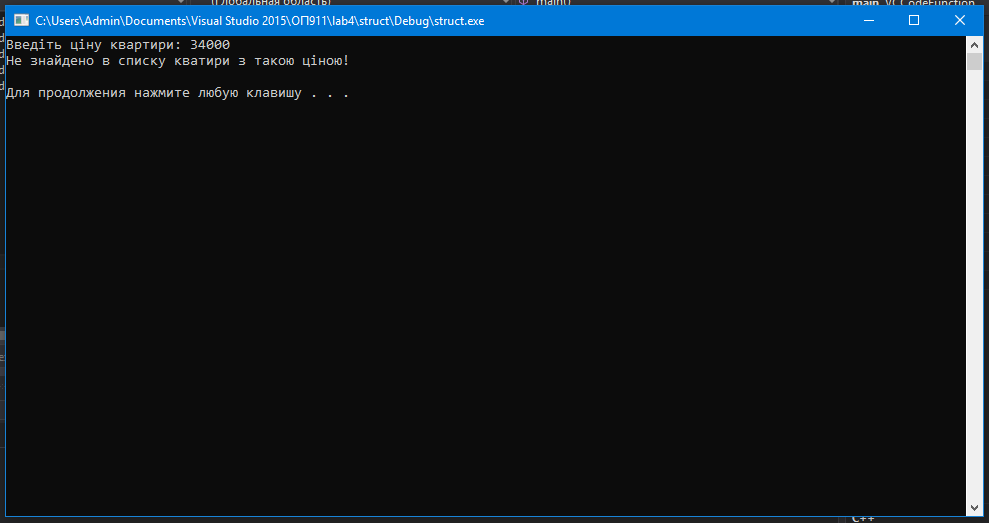
4)



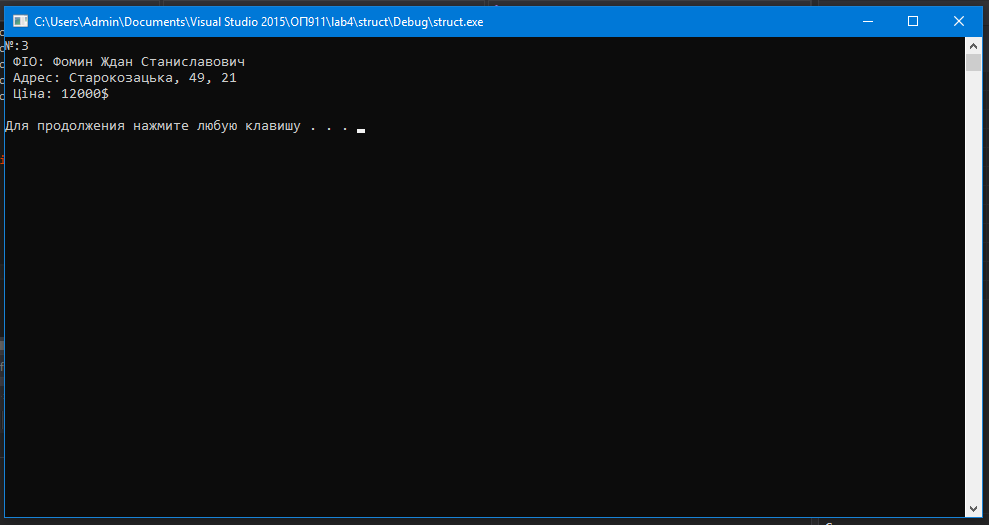
5)



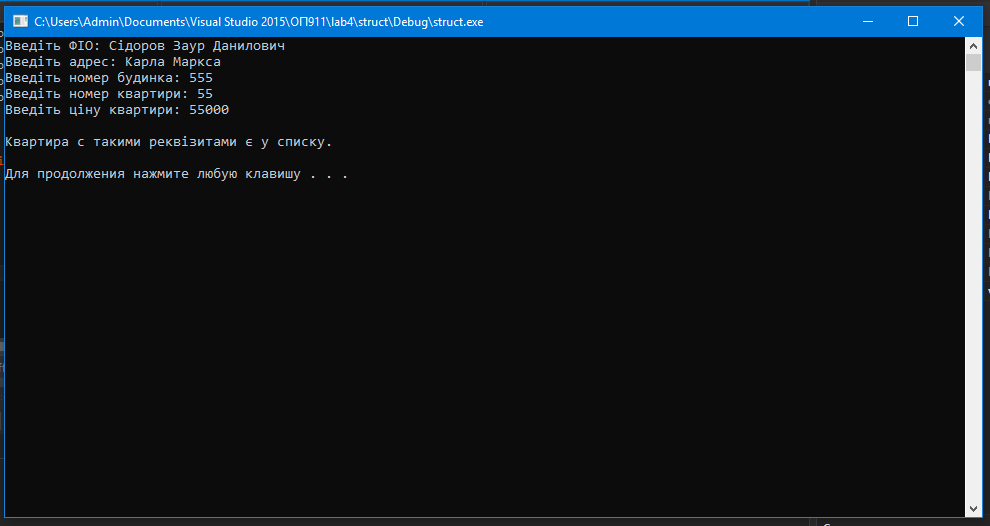
6)



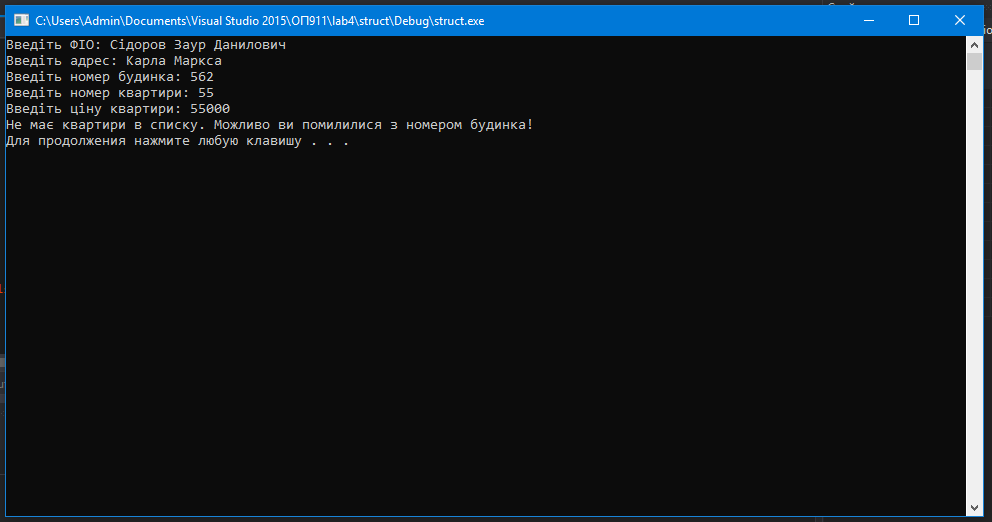
7)



8)



9)



Висновок: функція в програмуванні - фрагмент програмного коду (підпрограма), до якого можна звернутися з іншого місця програми. Функції дозволяють скоротати код, якщо якісь дії повторюються їх можна записати в функцію, і в подальшому викликати її с потрібними параметрами.

Багатофайлові програми дозволяють більш зрозуміліше читати код та його розуміти. Також над якоюсь частиною програми може працювати один програміст, а над іншою другий.