Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра ЕОМ

**ЛАБОРАТОРНА РОБОТА № 5.**

**Перевантаження операторів**

**Варіант - 17**

Виконав: ст.гр. КІ-15

Леземезюк Т.Т.

Прийняв: асистент

Козак Н.Б.

Львів – 2020

***Мета: познайомитися із перевантаженням операторів.***

***ТЕОРЕТИЧНІ ВІДОМОСТІ***

***Перевантаження операторів***

***Кожному оператору мова С++ ставить у відповідність ім'я функції, що складається з ключового слова operator, власне оператору та аргументів відповідних типів:***

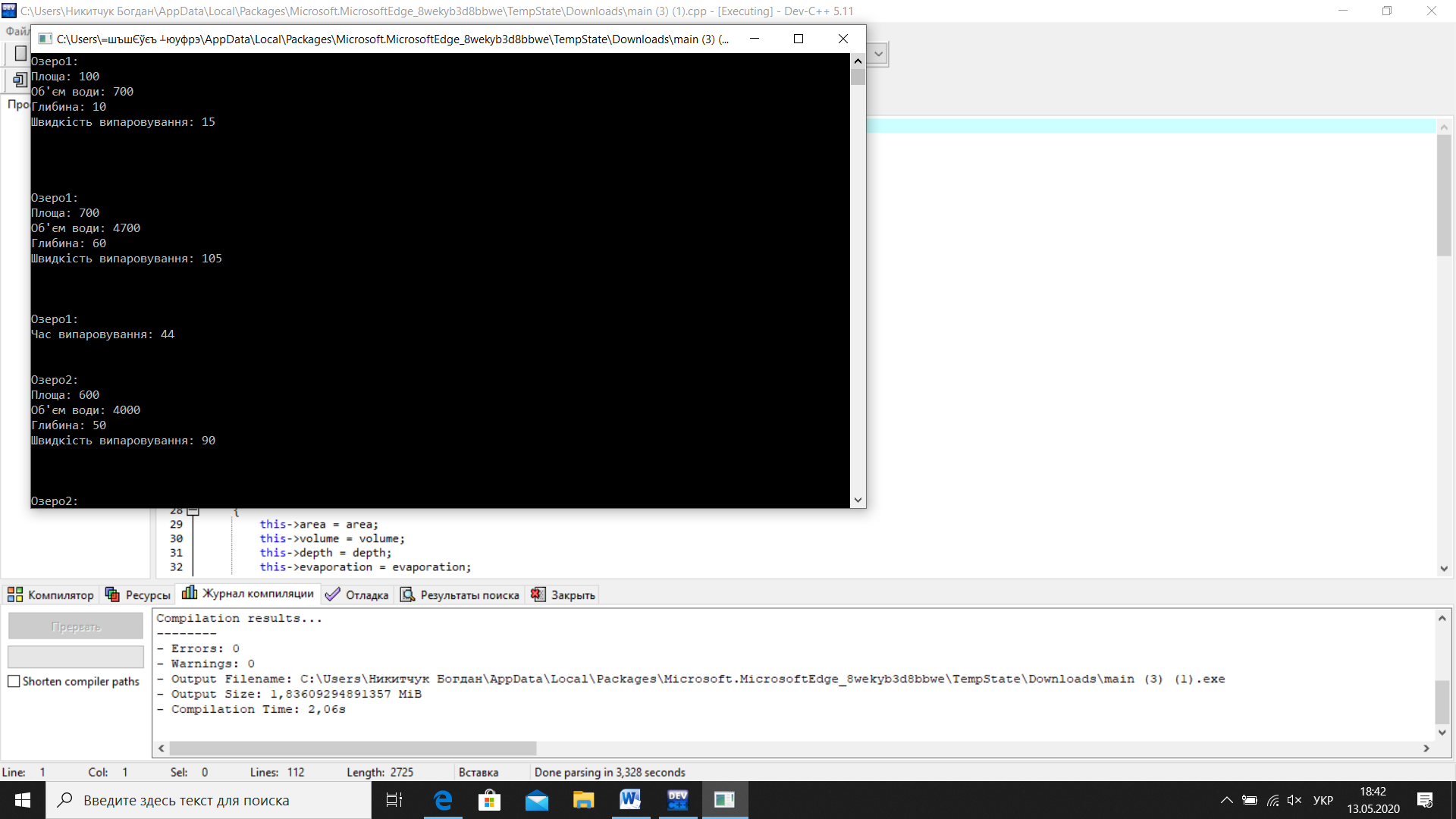
***тип operator символОператору (списокПараметрів) { //тіло методу }***

***Щоб використовувати операцію над об'єктами класів, ця операція повинна бути перевантажена, але є два виключення. Операції присвоювання (=) і взяття адреси (&) створюються в класі автоматично за замовчуванням, тому їх можна використовувати без явного перевантаження. За замовчуванням операція присвоювання зводиться до побітового копіювання даних-елементів класу. Проте таке побітове копіювання небезпечне для класів з елементами, що вказують на динамічно виділені області пам'яті, масиви, рядки, оскільки в цьому випадку відбувається копіювання не даних (глибоке копіювання), а лише вказівників на дані (поверхневе копіювання). Для таких класів слід явно перевантажувати операцію присвоювання і здійснювати у ній глибоке копіювання. Операція адресації також може бути використана з об'єктами будь-яких класів без перевантаження. Вона просто повертає адресу об'єкта в пам'яті. Але операцію адресації можна також і перевантажувати. Перевантаження операцій підпорядковується наступним правилам:***

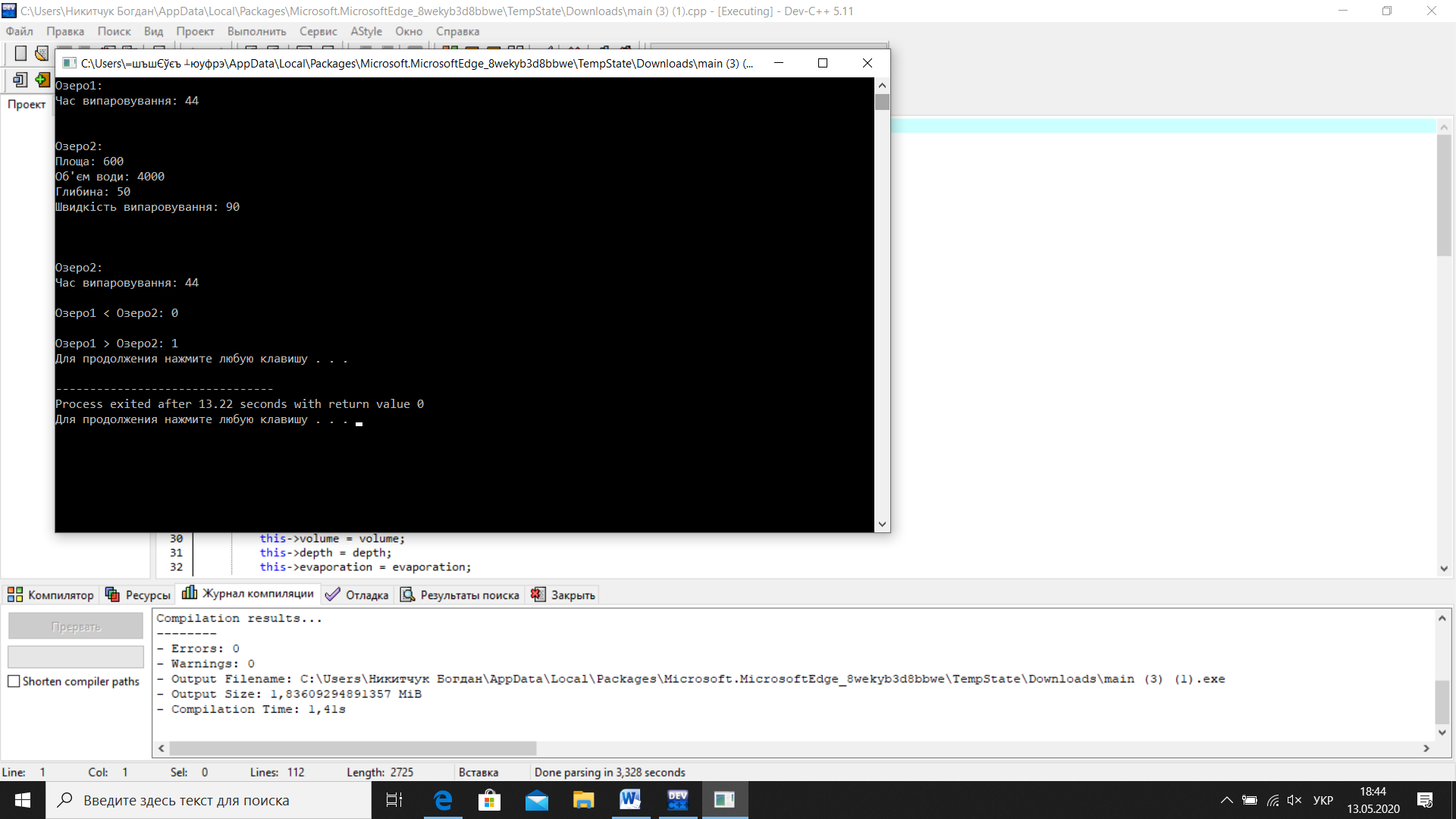
* ***При перевантаженні зберігаються кількість аргументів, пріоритети операцій та правила асоціації, що використовуються у стандартних типах даних;***
* ***Для стандартних типів даних операції не підлягають перевизначенню;***
* ***Перевантажена функція-оператор не може мати параметрів по замовчуванню, не успадковується та не може бути визначеною як static;***
* ***Функція-оператор може бути визначена трьома способами – метод класу, дружня функція або звичайна функція. В останніх двох випадках вона повинна приймати хоча б один аргумент, що має тип класу, вказівника або посилання на клас. При перевантаженні операцій ( ), [], -> та = функція перевантаження операції може бути оголошена лише як метод класу. Для інших операцій функції перевантаження операцій можуть не бути методами класу.***

***Результат виконання:***

**Скрін 1**



**Скрін 2**



**Код програми**

**#include <iostream>**

**#include <iomanip>**

**#include <Windows.h>**

**using namespace std;**

**class CLake**

**{**

**private:**

**int area,**

**volume,**

**depth,**

**evaporation;**

**public:**

**CLake(int area, int volume, int depth, int evaporation)**

**{**

**setstate(area, volume, depth, evaporation);**

**}**

**void fill(int amount)**

**{**

**volume += amount;**

**}**

**void dryingTime()**

**{**

**cout << "Час випаровування: " << volume / evaporation << endl;**

**}**

**void setstate(int area, int volume, int depth, int evaporation)**

**{**

**this->area = area;**

**this->volume = volume;**

**this->depth = depth;**

**this->evaporation = evaporation;**

**}**

**void currentstate()**

**{**

**cout << "Площа: " << area << endl;**

**cout << "Об'єм води: " << volume << endl;**

**cout << "Глибина: " << depth << endl;**

**cout << "Швидкість випаровування: " << evaporation << endl;**

**}**

**CLake operator + (const CLake& Lake)**

**{**

**area = area + Lake.area;**

**volume = volume + Lake.volume;**

**depth = depth + Lake.depth;**

**evaporation = evaporation + Lake.evaporation;**

**return \*this;**

**}**

**CLake& operator -= (const CLake& Lake)**

**{**

**area -= Lake.area;**

**volume -= Lake.volume;**

**depth -= Lake.depth;**

**evaporation -= Lake.evaporation;**

**return \*this;**

**}**

**bool operator < (CLake& Lake)**

**{**

**return volume < Lake.volume;**

**};**

**bool operator > (CLake& Lake)**

**{**

**return volume > Lake.volume;**

**};**

**friend ostream& operator << (ostream& os, const CLake& Lake) {**

**cout << "Площа: " << Lake.area << endl;**

**cout << "Об'єм води: " << Lake.volume << endl;**

**cout << "Глибина: " << Lake.depth << endl;**

**cout << "Швидкість випаровування: " << Lake.evaporation << endl;**

**os << endl;**

**return os;**

**}**

**friend istream& operator >> (istream& is, CLake& Lake) {**

**cout << "Введіть площу: " << endl;**

**is >> Lake.area;**

**cout << "Введіть об'єм води: " << endl;**

**cin >> Lake.volume;**

**cout << "Введіть глибину: " << endl;**

**is >> Lake.depth;**

**cout << "Введіть швидкість випаровування води: " << endl;**

**is >> Lake.evaporation;**

**return is;**

**}**

**};**

**int main()**

**{**

**SetConsoleCP(1251);**

**SetConsoleOutputCP(1251);**

**CLake Lake1(100, 700, 10, 15), Lake2(600, 4000, 50, 90);**

**cout << "Озеро1: " << endl; cout << Lake1 << endl;**

**cout << endl;**

**Lake1 = Lake1 + Lake2;**

**cout << endl;**

**cout << "Озеро1: " << endl; cout << Lake1 << endl;**

**cout << endl;**

**cout << "Озеро1: " << endl; Lake1.dryingTime();**

**cout << endl;**

**cout << endl;**

**cout << "Озеро2: " << endl; cout << Lake2 << endl;**

**cout << endl;**

**cout << "Озеро2: " << endl; Lake2.dryingTime();**

**cout << endl;**

**int k = Lake1 < Lake2;**

**cout << "Озеро1 < Озеро2: " << k << endl;**

**cout << endl;**

**k = Lake1 > Lake2;**

**cout << "Озеро1 > Озеро2: " << k << endl;**

**system("pause");**

**return 0;**

**}**