Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра ЕОМ

**ЛАБОРАТОРНА РОБОТА № 4. КЛАСИ ТА ОБ’ЄКТИ**

Виконав: ст.гр. КІ-16

Леземезюк Т.Т.

Прийняв: асистент

Козак Н.Б.

Львів – 2020

***Мета: познайомитися із класами та об‘єктами.***

***КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ***

***Визначення класу***

***Основна відмінність будь-якої об‘єктно-орієнтованої мови програмування від інших не об‘єктно-орієнтованих мов програмування є можливість створення нових визначених користувачем типів, що називаються класами. Клас – це визначений користувачем тип з даними-елементами (властивостями) та функціями-елементами (методами), що являються членами класу. Він зазвичай описує певну абстракцію якоїсь сутності реального світу з її властивостями і можливими діями. Змінна типу клас називається об‘єктом. Об‘єкт – це вже не абстракція, а елемент реального світу, який може володіти певними характеристиками, які задаються властивостями в класі, та діяти згідно операцій заданих методами у класі. Оголошення класу в мові С++ має наступний синтаксис (не обов‘язково щоб клас мав всі секції чи спадкував базовий клас):***

***class ім'я\_класу : список\_базових\_класів ,***

***publіc: // Оголошення загальнодоступних (відкритих) членів класу, що можуть бути доступні звідусіль protected: // Оголошення захищених членів класу, що доступні тільки для похідних класів prіvate: // Оголошення закритих членів класу, що доступні лише членам цього класу, та не можуть бути змінені чи викликані з-за меж класу напряму, а лише за посередництвом методів з інших секцій };***

***Члени класу (властивості і методи), оголошені після ключового слова publіc стають відкритими членами. Це означає, що вони доступні:***

* ***усім іншим членам класу;***
* ***дружнім конструкціям (класам, методам, функціям);***
* ***членам похідних класів;***
* ***з-під об‘єктів класу після створення об‘єктів; після створення об‘єкту класу до його членів можна звертатися лише якщо вони є загальнодоступними.***
* ***Члени класу, оголошені після ключового слова prіvate, стають закритими членами. Вони доступні: для інших членів того ж класу;***
* ***друзям класу. Якщо в класі не оголошено секцій, але визначено методи чи властивості, то вони вважатимуться такими, що оголошені у секції prіvate. Для модифікації властивостей, що оголошені в секції private створюються спеціальні методи, які здійснюють цю модифікацію. Назви таких методів прийнято починати зі слова set. На жаргоні такі методи інколи називають сетерами. Для читання властивостей, що оголошені в секції private створюються спеціальні методи, які безпечно повертають значення цих властивостей. Назви таких методів прийнято починати зі слова get. На жаргоні такі методи інколи називають ґетерами.***

***Члени класу, оголошені після ключового слова protected, стають захищеними членами. Вони доступні:***

* ***для інших членів того ж класу;***
* ***членам похідних класів;***
* ***друзям класу.***

***Рівні доступу членів класу Доступ private protected public Інші члени цього класу + + + Друзі класу + + + Члени похідних класів - + + З-під об‘єктів класу - - +***

***Оголошення класу містить оголошення даних-елементів (властивостей) та функцій-елементів (методів) класу. Одна з фундаментальних основ ООП передбачає інкапсуляцію даних, тобто дані мають бути недоступними ззовні, а***

***лише за посередництвом загальнодоступних методів класу. Оголошення методів має вигляд оголошення прототипу функції в середині однієї з секцій класу, зазвичай у секції public. Якщо методи є службовими і не мають бути доступні ззовні, тоді їх розміщують в секції private або protected. Сама ж реалізація методу може знаходитись як усередині класу (тоді оголошення методу в класі має вигляд оголошення функції з тілом), так і поза оголошенням класу (тоді у класі оголошується прототип функції, а її тіло визначається за межами класу). Але один з найфундаментальніших принципів розробки програмного забезпечення мовою С++ полягає у розмежуванні інтерфейсу класу від його реалізації. Тому при побудові програми мовою C++ кожне оголошення класу зазвичай розміщують у заголовочний файл \*.h назва якого співпадає з назвою класу, а реалізацію методів цього класу – у файл \*.cpp із тим ж іменем, що й \*.h файл. Заголовочні файли включаються (за допомогою директиви #іnclude) у кожен файл, у якому використовується клас, а файли з вихідними кодами компілюються і компонуються з файлом, що містить головну програму (main-функцію). При визначенні методу класу за межами класу необхідно зв'язати ім'я відповідного методу з ім'ям класу. Зв'язок встановлюється шляхом написання імені класу, оператора розширення області видимості (::) та імені методу. Оскільки різні класи можуть мати елементи з однаковими іменами, то оператор розширення області видимості дозволяє однозначно ідентифікувати методи конкретного класу.***

***тип ім’яКласу :: ім’яМетоду (списокПараметрів) { // тіло методу }***

***Областю видимості властивостей та методів класу є клас, тобто все що оголошене в середині класу є видиме і доступне за іменем елементів в середині класу без додаткових маніпуляцій.***

**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;**

**}**

**};**

**int main()**

**{**

**SetConsoleCP(1251);**

**SetConsoleOutputCP(1251);**

**CLake Lake(500, 700, 20, 30);**

**Lake.currentstate();**

**Lake.fill(50);**

**cout << endl;**

**Lake.currentstate();**

**cout << endl;**

**Lake.dryingTime();**

**system("pause");**

**return 0;**

**}**