Лабораторна робота №1

Звіт

З дисципліни “Об'єктно-орієнтоване програмування ”

на тему: “Знайомство з класами та об’єктами ”.

Студента 3 курсу: Групи МІТ-31 Демиденко Андрій

**Київ - 2024р.**

**Комплексний звіт**

**Мета:** ознайомитися з парадигмою ООП, розглянути поняття класу та об’єкта, навчитись створювати класи та об’єкти в Пайтон

**Завдання для самостійного опрацювання варіант**

**Варіант №5**

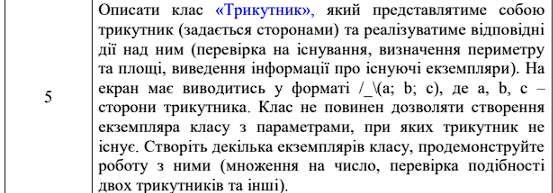


Рис. 1.1 – завдання для варіанту

**Завдання №1**

Для початку слід розібрати завдання відповідно до варіанту (рис. 1.1). Проаналізувавши завдання, можна уявити структуру класу та його методів, і реалізувати їх відповідно до умов завдання. Нижче будуть наведені рисунки з 1.2 по 1.10 з реалізацією в коді. Для кращого розгляду кожного блоку коду в цьому абзаці буде описано що кожен метод реалізує та його особливості.

**Метод \_\_init\_\_(self, a, b, c)** – це приклад магічного методу в Пайтон який викликається автоматично після створення об’єкта, для ініціалізації атрибутів об’єктів (рис. 1.2).

**Метод perimeter(self) –** рахує периметр для трикутника шляхом додавання значень атрибутів екземпляру і повертає відповідне значення (рис. 1.3).

**Метод square(self) –** рахує площу трикутника шляхом виконання математичної формули та повертає отриманий результат (рис. 1.4).

**Метод** **similarity(self, other)** – порівнює 2 трикутники використовуючи теорему про подібність за співвідношення 3х сторін (рис. 1.5).

**Метод \_\_str\_\_(self) –** магічний метод який виводить інформацію про трикутник, перевизначає str() (рис. 1.6).

**Метод \_\_gt\_\_(self, other) –** перевизначає метод < (рис. 1.7).

**Метод \_\_lt\_\_(self, other) –** перевизначає метод > (рис. 1.8).

**Метод \_\_mul\_\_(self, x) –** специфічним чином перевизначає метод \* об’єкту класу на число (рис. 1.9).

**Метод \_\_truediv\_\_(self, x) –** специфічним чином перевизначає метод / об’єкту класу на число (рис. 1.10).

**Метод display\_all\_triangles() –** це статичний метод, його особливістю є те що він не приймає параметр serlf і не має доступу до атрибутів екземпляру. Замість цього він працює з атрибутами класу і може бути викликаний без створення екземпляра класу, позначається відповідним декоратором @staticmethod. Також він взаємодіє з атрибутом класу list\_of\_items (рис. 1.11).

Всі магічні методи крім \_\_str\_\_ та \_\_init\_\_. Використовують перевірку значень, що вводяться, для коректної роботи з об’єктами.

Після цього створюються об’єкти класу в тілі програми і викликається кожен метод класу для реалізації (рис. 1.12).

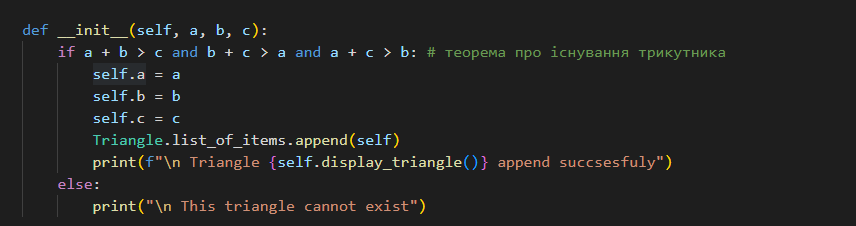


Рис. 1.2 – Метод \_\_init\_\_(self, a, b, c)

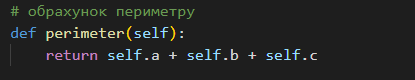


Рис. 1.3 – Метод scale(self, x), Метод perimeter(self)

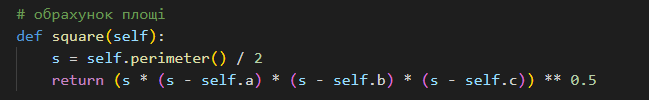


Рис. 1.4 – Метод square(self)

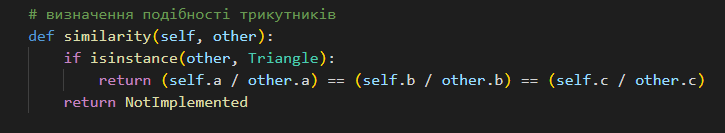


Рис. 1.5 – Метод similarity(self, other)

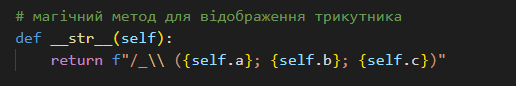


Рис. 1.6 – Метод \_\_str\_\_(self)

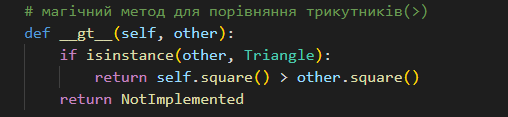


Рис. 1.7 – Метод \_\_gt\_\_(self, other)

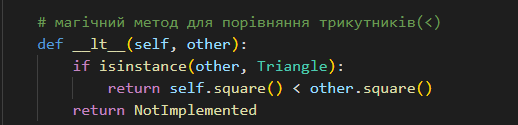


Рис. 1.8 – Метод \_\_lt\_\_(self, other)

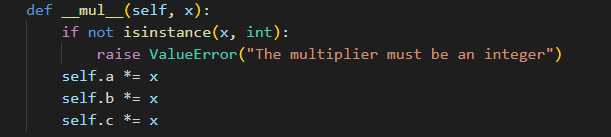


Рис. 1.9 – Метод \_\_mul\_\_(self, x)

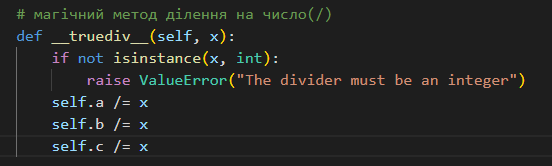


Рис. 1.10 – Метод \_\_truediv\_\_(self, x)

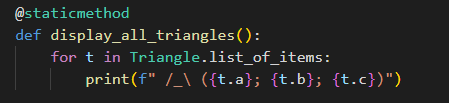


Рис. 1.11 – Метод display\_all\_triangles()

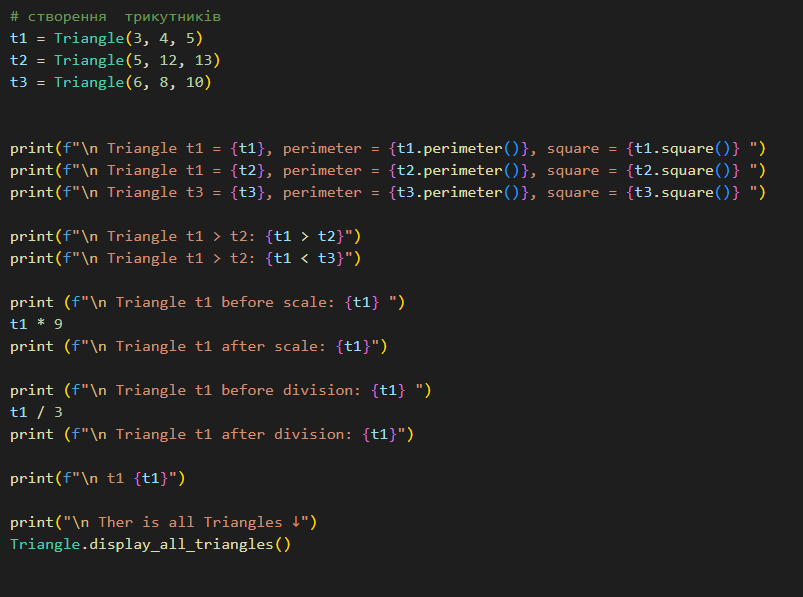


Рис. 1.12 – Реалізація програми

**Завдання №2**

Тепер для виконання наступного завдання потрібно реалізувати програму з подібним функціоналом на іншій мові програмування що підтримую ООП. Обраною мовою стала С++ через вже наявний досвід з програмування на ній. Оскільки клас реалізований з тими самим функції то в цьому блоці буде розглянута лише відмінність від Python.

Крім синтаксису, основні відмінності між Python і С++, реалізація статичних методів, керування пам’яттю, безпека. А саме, пам’ять для статичного атрибуту класу виділяється в тілі програми, а не в класі (це використовується через те що береться <vector> а не простий тип даних по типу int, str…), існує this що є неявним вказівником і використовується для розрізнення локальних змінних та змінних класу, а також для відображення поточного об’єкту методу. Також щоб не змінювати атрибутів класу використовується const, а приватність оглошується вже після всіх методів класу, а не під час ініціалізації, як в Python. Код класу (рис. 1.13), код основної програми (рис. 1.14).

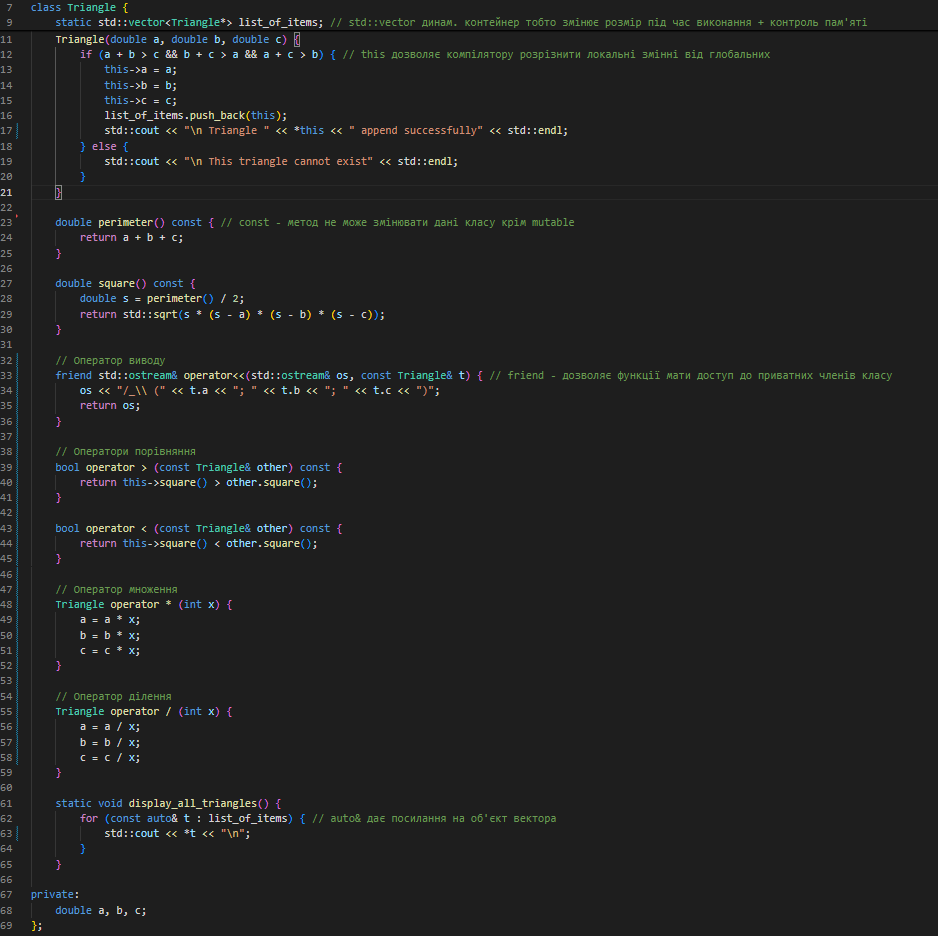


Рис. 1.13 – реалізація класу на С++

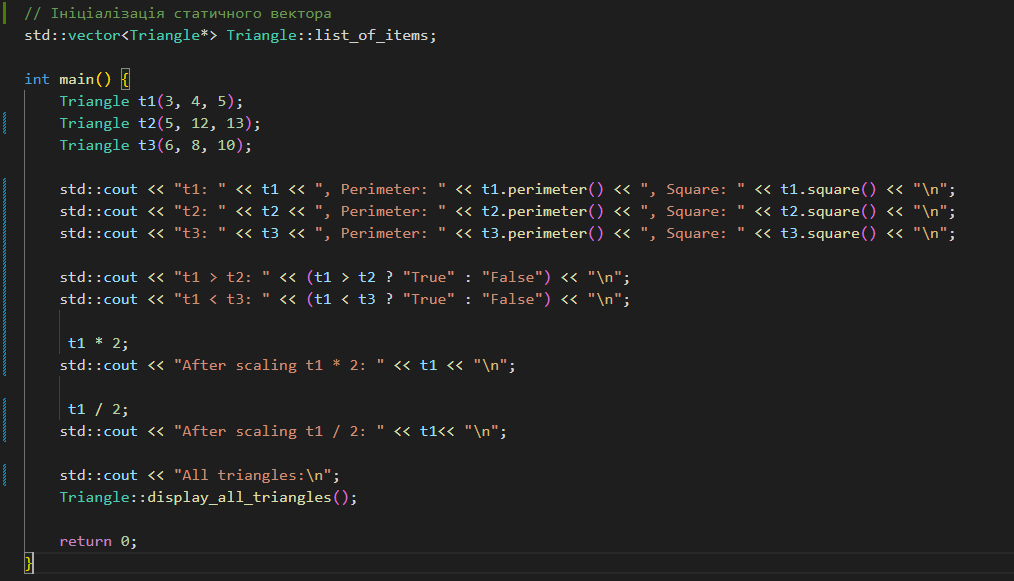


Рис. 1.14 – використання методів класу на С++

**Висновок:** Досліджено дуже базові можливості класу на прикладі виконання програми на Python та C++. Python здався більш грубим, але простим у використанні, С++ навпаки більш складний і потребує набагато більше уваги в контролі пам’яті, але задався більш простим для розуміння особливостей ООП. Тому на мою думку C++ є більш підходящою мовою в парадигмі використання принципів ООП, але оскільки Python на даний момент є більш універсальним і простим, та славиться гнучкістю, надаю перевагу саме йому.

[Git Reposytory](https://github.com/AndrewDem0/OOP-Learning)