Міністерство освіти і науки України

Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії

Кафедра програмної інженерії

ЗВІТ

до практичної роботи №2.2

з навчальної дисципліни «Об'єктно-орієнтоване програмування»

**Тема**: «Мультифайлові програми»

Підготував:

студент групи СПзс-21

Карпюк Олександра Андріївна

Варіант №1

Тернопіль 2024

**Мета**: Метою лабораторної роботи є вивчення та практичне застосування принципів мультифайлової організації коду, об'єктно-орієнтованого програмування, шаблонів функцій та класів, а також робота з алгоритмами та контейнерами стандартної бібліотеки C++.

**Завдання**

Варіант 1: Бібліотека для роботи з Геометричними фігурами.

1. Створити бібліотеку класів:

* Реалізувати класи для геометричних фігур (коло, прямокутник, трикутник).
* Забезпечити можливість обчислення площі та периметру кожної фігури.

2. Реалізувати клієнтський додаток:

* Використовуючи бібліотеку, створити клієнтський додаток для взаємодії з геометричними фігурами.
* Додаток має демонструвати використання розробленої бібліотеки для обчислення площі та периметру.

3. Тестування програми:

* Створити main() або клієнтську програму для тестування.
* Провести тестування розробленої та реалізованої програми з використанням мультифайлової організації коду та ООП-підходу.

**Хід роботи**

Клас geometry.h

#pragma once

#ifndef GEOMETRY\_H

#define GEOMETRY\_H

#include <string>

#include <cmath>

// Абстрактний клас для геометричних фігур

class Shape {

public:

virtual double getArea() const = 0; // Обчислення площі

virtual double getPerimeter() const = 0; // Обчислення периметру

virtual std::string getName() const = 0; // Отримати назву фігури

virtual ~Shape() = default;

};

// Клас для кола

class Circle : public Shape {

private:

double radius;

public:

Circle(double r);

double getArea() const override;

double getPerimeter() const override;

std::string getName() const override;

};

// Клас для прямокутника

class Rectangle : public Shape {

private:

double width, height;

public:

Rectangle(double w, double h);

double getArea() const override;

double getPerimeter() const override;

std::string getName() const override;

};

// Клас для трикутника

class Triangle : public Shape {

private:

double a, b, c; // Сторони трикутника

public:

Triangle(double side1, double side2, double side3);

double getArea() const override;

double getPerimeter() const override;

std::string getName() const override;

};

#endif // GEOMETRY\_H

Клас geometry.cpp

#define \_USE\_MATH\_DEFINES

#include "geometry.h"

#include <cmath>

Circle::Circle(double r) : radius(r) {}

double Circle::getArea() const {

return M\_PI \* radius \* radius;

}

double Circle::getPerimeter() const {

return 2 \* M\_PI \* radius;

}

std::string Circle::getName() const {

return "Circle";

}

Rectangle::Rectangle(double w, double h) : width(w), height(h) {}

double Rectangle::getArea() const {

return width \* height;

}

double Rectangle::getPerimeter() const {

return 2 \* (width + height);

}

std::string Rectangle::getName() const {

return "Rectangle";

}

Triangle::Triangle(double side1, double side2, double side3) : a(side1), b(side2), c(side3) {}

double Triangle::getArea() const {

double s = getPerimeter() / 2; // Півпериметр

return std::sqrt(s \* (s - a) \* (s - b) \* (s - c)); // Формула Герона

}

double Triangle::getPerimeter() const {

return a + b + c;

}

std::string Triangle::getName() const {

return "Triangle";

}

Код main (GeometryLab):

#include <iostream>

#include <vector>

#include <memory>

#include "geometry.h"

int main() {

// Вектор для зберігання фігур

std::vector<std::shared\_ptr<Shape>> shapes;

// Додавання різних фігур

shapes.push\_back(std::make\_shared<Circle>(5.0));

shapes.push\_back(std::make\_shared<Rectangle>(4.0, 6.0));

shapes.push\_back(std::make\_shared<Triangle>(3.0, 4.0, 5.0));

// Вивід інформації про фігури

for (const auto& shape : shapes) {

std::cout << "Shape: " << shape->getName() << std::endl;

std::cout << "Area: " << shape->getArea() << std::endl;

std::cout << "Perimeter: " << shape->getPerimeter() << std::endl;

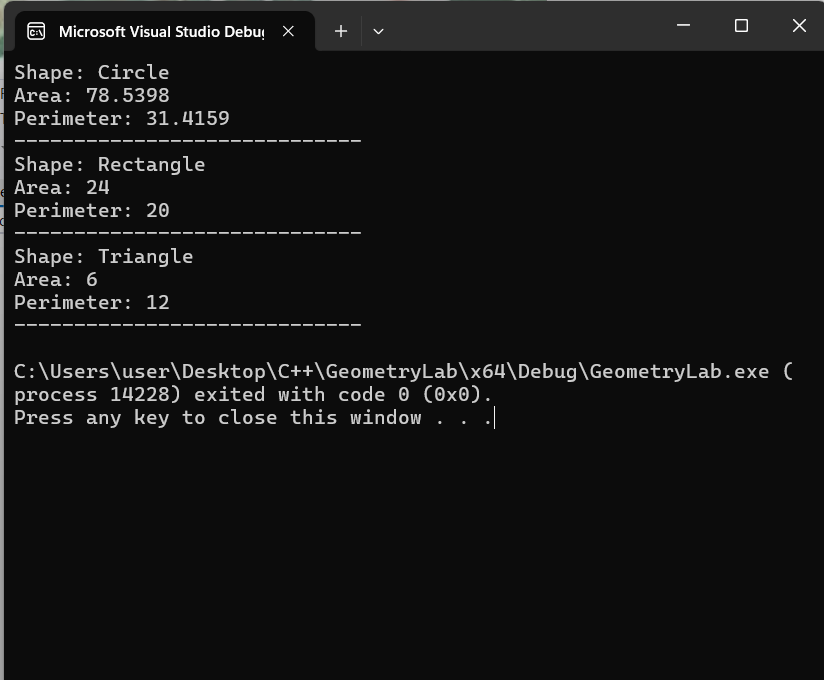
std::cout << "-----------------------------" << std::endl;

}

return 0;

}

Результат виконання:



**Висновки**:.

У ході виконання даної лабораторної роботи я закріпила знання з об'єктно-орієнтованого програмування (ООП) та застосувала принципи мультифайлової організації коду. Основні результати роботи:

1. **Вивчення ООП**:
   * Навчилася створювати абстрактні базові класи та реалізовувати їх наслідування.
   * Застосувала поліморфізм для роботи з різними геометричними фігурами, реалізувавши спільний інтерфейс.
2. **Мультифайлова організація коду**:
   * Окремо описала заголовки класів у файлі geometry.h і реалізувала їх у geometry.cpp, що покращує читабельність і модульність коду.
   * Створила клієнтський код (main.cpp), який демонструє використання створених класів.
3. **Робота з шаблонами та стандартною бібліотекою C++**:
   * Використала шаблони функцій і стандартний контейнер std::vector для зберігання об'єктів різних типів.
   * Застосувала алгоритми стандартної бібліотеки для роботи з контейнерами та обчислення властивостей об'єктів.
4. **Тестування програми**:
   * Створила програму для обчислення площі та периметра геометричних фігур (коло, прямокутник, трикутник).
   * Перевірила коректність роботи програми з використанням тестових прикладів.

У результаті роботи я покращила свої навички програмування на C++, зокрема:

* Розуміння структури та організації складних проектів.
* Використання принципів ООП у практичних задачах.
* Роботу зі стандартними контейнерами STL та реалізацію клієнтського додатку.

Ця лабораторна робота допомогла мені закріпити теоретичні знання та відчути практичну користь від грамотної організації коду, що є основою для створення складних програмних систем.