**Лабораторна робота №5.**

**Тема:** *Спадкування. Ієрархія та композиція.*

**Методичні вказівки**

**Завдання 1**

Задача 1.14. Створити клас Triad (трійка чисел); визначити методи зміни полів і обчислення суми чисел. Визначити похідний клас Triangle з полями сторін. Визначити методи обчислення кутів і площі трикутника.

**Код:**

**#include <iostream>**

**#include <cmath>**

**using namespace std;**

**class Triad {**

**protected:**

**int a, b, c;**

**public:**

**Triad(int x = 0, int y = 0, int z = 0) : a(x), b(y), c(z) {**

**cout << "Створено об'єкт Triad\n";**

**}**

**~Triad() {**

**cout << "Знищено об'єкт Triad\n";**

**}**

**void setValues(int x, int y, int z) {**

**a = x; b = y; c = z;**

**}**

**int sum() const {**

**return a + b + c;**

**}**

**void print() const {**

**cout << "Трійка: " << a << ", " << b << ", " << c << endl;**

**}**

**};**

**class Triangle : public Triad {**

**public:**

**Triangle(int x = 0, int y = 0, int z = 0) : Triad(x, y, z) {**

**cout << "Створено об'єкт Triangle\n";**

**}**

**~Triangle() {**

**cout << "Знищено об'єкт Triangle\n";**

**}**

**bool isValid() const {**

**return (a + b > c && a + c > b && b + c > a);**

**}**

**double area() const {**

**if (!isValid()) return 0;**

**double p = sum() / 2.0;**

**return sqrt(p \* (p - a) \* (p - b) \* (p - c));**

**}**

**void angles() const {**

**if (!isValid()) {**

**cout << "Некоректний трикутник\n";**

**return;**

**}**

**double alpha = acos((b\*b + c\*c - a\*a) / (2.0 \* b \* c)) \* 180 / M\_PI;**

**double beta = acos((a\*a + c\*c - b\*b) / (2.0 \* a \* c)) \* 180 / M\_PI;**

**double gamma = 180.0 - alpha - beta;**

**cout << "Кути трикутника: " << alpha << ", " << beta << ", " << gamma << endl;**

**}**

**};**

**int main() {**

**Triangle t(3, 4, 5);**

**t.print();**

**cout << "Сума: " << t.sum() << endl;**

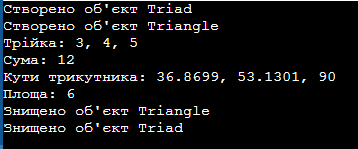
**t.angles();**

**cout << "Площа: " << t.area() << endl;**

**return 0;**

**}**

**Скрін:**



**Завдання 2**

Задача 2.14. Створити клас вікно, використовуючи композицію й ієрархію, визначити вікно із заголовком і кнопкою. Клас заголовок і кнопка теж повинні бути визначені. Навісити на кнопку функцію зворотного виклику - (метод класу вікна). Ця функція викликається, коли натискається будь-яка клавіша.

**Код:**

**#include <iostream>**

**#include <string>**

**using namespace std;**

**// Базовий клас Вікно**

**class Window {**

**public:**

**virtual void show() const = 0;**

**virtual void onButtonPressed() {**

**cout << "Callback: Кнопка натиснута!" << endl;**

**}**

**virtual ~Window() {}**

**};**

**// Клас Заголовка**

**class Title {**

**string text;**

**public:**

**Title(const string& t) : text(t) {}**

**void show() const {**

**cout << "Заголовок: " << text << endl;**

**}**

**};**

**// Клас Кнопки**

**class Button {**

**string label;**

**Window\* parent;**

**public:**

**Button(const string& lbl, Window\* wnd) : label(lbl), parent(wnd) {}**

**void press() {**

**if (parent) parent->onButtonPressed();**

**}**

**void show() const {**

**cout << "[Кнопка: " << label << "]" << endl;**

**}**

**};**

**// Похідний клас із заголовком**

**class TitleWindow : public Window {**

**protected:**

**Title title;**

**public:**

**TitleWindow(const string& t) : title(t) {}**

**void show() const override {**

**cout << "=== Вікно з заголовком ===" << endl;**

**title.show();**

**}**

**};**

**// Похідний клас із кнопкою**

**class ButtonWindow : public TitleWindow {**

**Button button;**

**public:**

**ButtonWindow(const string& t, const string& btnLabel)**

**: TitleWindow(t), button(btnLabel, this) {}**

**void show() const override {**

**TitleWindow::show();**

**button.show();**

**}**

**void simulateButtonPress() {**

**button.press();**

**}**

**};**

**int main() {**

**ButtonWindow win("Головне вікно", "ОК");**

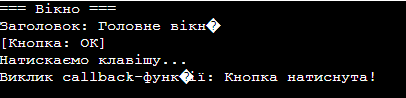
**win.show();**

**cout << "Натискаємо кнопку..." << endl;**

**win.simulateButtonPress();**

**return 0;**

**}**

**Скрін:  
**

**Завдання 3**

Задача 3.14. Створити ієрархію класів рядків і рядків із прописних букв. Перевизначити вивід у потік і введення з потоку, конструктор копіювання, оператор присвоювання через відповідні функції базового класу.

**Код:  
#include <iostream>**

**#include <string>**

**#include <algorithm>**

**using namespace std;**

**class MyString {**

**protected:**

**string data;**

**public:**

**MyString(const string& str = "") : data(str) {}**

**MyString(const MyString& other) {**

**data = other.data;**

**}**

**MyString& operator=(const MyString& other) {**

**if (this != &other)**

**data = other.data;**

**return \*this;**

**}**

**virtual void set(const string& str) {**

**data = str;**

**}**

**virtual void print(ostream& os) const {**

**os << data;**

**}**

**virtual void read(istream& is) {**

**getline(is, data);**

**}**

**friend ostream& operator<<(ostream& os, const MyString& s) {**

**s.print(os);**

**return os;**

**}**

**friend istream& operator>>(istream& is, MyString& s) {**

**s.read(is);**

**return is;**

**}**

**};**

**class UpperString : public MyString {**

**public:**

**UpperString(const string& str = "") {**

**set(str);**

**}**

**UpperString(const UpperString& other) : MyString(other) {}**

**UpperString& operator=(const UpperString& other) {**

**if (this != &other)**

**MyString::operator=(other);**

**return \*this;**

**}**

**void set(const string& str) override {**

**data = str;**

**transform(data.begin(), data.end(), data.begin(), ::toupper);**

**}**

**void read(istream& is) override {**

**getline(is, data);**

**transform(data.begin(), data.end(), data.begin(), ::toupper);**

**}**

**};**

**int main() {**

**MyString s1;**

**cout << "Введіть рядок: ";**

**cin.ignore(); // щоб очистити буфер після попереднього вводу (особливо у Visual Studio)**

**cin >> s1;**

**cout << "Ви ввели: " << s1 << endl;**

**UpperString s2;**

**cout << "Введіть рядок (буде переведено у верхній регістр): ";**

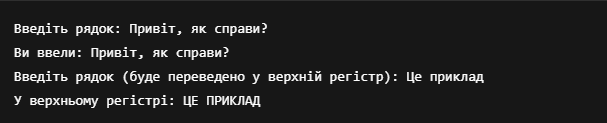
**cin >> s2;**

**cout << "У верхньому регістрі: " << s2 << endl;**

**return 0;**

**}**

**Скрін:**

****