МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 4

з навчальної дисципліни

# “ ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ ”

Поліморфізм. Обробка винятків

ВИКОНАВ

студент академічної групи КБ-23

Сергій Козирь

ПЕРЕВІРИВ

Викладач

Козірова Н.Л.

Кропивницький – 2024

ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Лабораторна робота №4

Поліморфізм. Обробка винятків

Мета: Ознайомитись з поняттям поліморфізму у мові C++ та навчитись використовувати віртуальні функції для досягнення поліморфізму. Також вивчити принципи обробки винятків у мові C++.

Варіант 7

1. Розробіть систему керування човнами, яка включатиме обробку різних типів човнів. Кожен тип човна має свої характеристики, які потрібно реалізувати за допомогою наслідування, сетерів та гетерів. Кожен човен має наступні характеристики:

• Назва

• Максимальна швидкість

• Вантажопідйомність Кожен тип човна має власні додаткові характеристики: Весловий човен:

• Кількість весел

• Тип матеріалу Моторний човен:

• Потужність двигуна

• Тип палива

2. Створіть базовий абстрактний клас Boat з віртуальними функціями та використати поліморфізм для реалізації додаткових методів та функцій. Також, додайте виняткові ситуації для обробки некоректних даних.

3. Створіть похідні класи Rowboat та Motorboat, які успадковуються від класу Boat. Реалізуйте в них відповідні віртуальні функції та додайте додаткові характеристики, які були зазначені вище.

4. У вашій програмі мають бути використані виняткові ситуації для обробки некоректних даних, наприклад, якщо некоректна швидкість човна або від'ємна вантажопідйомність. Ви можете розширити його, додати додаткові методи та функціональні можливості, які вам здаються відповідними.

Реалізація

#include <iostream>

#include <stdexcept>

#include <string>

using namespace std;

// Базовий абстрактний клас Boat

class Boat {

protected:

string name;

double maxSpeed;

double loadCapacity;

public:

// Конструктор

Boat(string boatName, double speed, double capacity)

: name(boatName), maxSpeed(speed), loadCapacity(capacity) {

if (maxSpeed < 0 || loadCapacity < 0) {

throw invalid\_argument("Некоректна швидкість або вантажопідйомність!");

}

}

// Віртуальні методи

virtual void showInfo() const = 0; // Віртуальна функція для відображення інформації

// Сетери

void setName(string boatName) { name = boatName; }

void setMaxSpeed(double speed) {

if (speed < 0) throw invalid\_argument("Швидкість не може бути від'ємною!");

maxSpeed = speed;

}

void setLoadCapacity(double capacity) {

if (capacity < 0) throw invalid\_argument("Вантажопідйомність не може бути від'ємною!");

loadCapacity = capacity;

}

// Гетери

string getName() const { return name; }

double getMaxSpeed() const { return maxSpeed; }

double getLoadCapacity() const { return loadCapacity; }

// Віртуальний деструктор

virtual ~Boat() {}

};

// Клас Rowboat (весловий човен)

class Rowboat : public Boat {

private:

int numberOfOars; // Кількість весел

string materialType; // Тип матеріалу

public:

Rowboat(string boatName, double speed, double capacity, int oars, string material)

: Boat(boatName, speed, capacity), numberOfOars(oars), materialType(material) {

if (numberOfOars <= 0) {

throw invalid\_argument("Кількість весел має бути більше нуля!");

}

}

// Сетери

void setNumberOfOars(int oars) {

if (oars <= 0) throw invalid\_argument("Кількість весел має бути більше нуля!");

numberOfOars = oars;

}

void setMaterialType(string material) { materialType = material; }

// Гетери

int getNumberOfOars() const { return numberOfOars; }

string getMaterialType() const { return materialType; }

// Перевизначення методу showInfo

void showInfo() const override {

cout << "Весловий човен: " << name << endl;

cout << "Максимальна швидкість: " << maxSpeed << " км/год" << endl;

cout << "Вантажопідйомність: " << loadCapacity << " кг" << endl;

cout << "Кількість весел: " << numberOfOars << endl;

cout << "Матеріал: " << materialType << endl;

}

};

// Клас Motorboat (моторний човен)

class Motorboat : public Boat {

private:

double enginePower; // Потужність двигуна

string fuelType; // Тип палива

public:

Motorboat(string boatName, double speed, double capacity, double power, string fuel)

: Boat(boatName, speed, capacity), enginePower(power), fuelType(fuel) {

if (enginePower <= 0) {

throw invalid\_argument("Потужність двигуна має бути більше нуля!");

}

}

// Сетери

void setEnginePower(double power) {

if (power <= 0) throw invalid\_argument("Потужність двигуна має бути більше нуля!");

enginePower = power;

}

void setFuelType(string fuel) { fuelType = fuel; }

// Гетери

double getEnginePower() const { return enginePower; }

string getFuelType() const { return fuelType; }

// Перевизначення методу showInfo

void showInfo() const override {

cout << "Моторний човен: " << name << endl;

cout << "Максимальна швидкість: " << maxSpeed << " км/год" << endl;

cout << "Вантажопідйомність: " << loadCapacity << " кг" << endl;

cout << "Потужність двигуна: " << enginePower << " к.с." << endl;

cout << "Тип палива: " << fuelType << endl;

}

};

int main() {

setlocale(LC\_ALL, "ukr");

try {

// Створюємо човни

Rowboat rowboat("Весловий Човен 1", 15.5, 200, 4, "Дерево");

Motorboat motorboat("Моторний Човен 1", 80.0, 500, 250, "Бензин");

// Виведення інформації про човни

rowboat.showInfo();

cout << endl;

motorboat.showInfo();

}

catch (const exception& e) {

cerr << "Помилка: " << e.what() << endl;

}

return 0;

}

