Лабораторна робота №4

Тема: Успадковування класів
Мета: ознайомитись зі способами та механізмами успадкування класів та навчитись використовувати їх для побудови об’єктно-орієнтованих програм.

Завдання 1. Уявіть собі видавничу компанію, яка торгує книгами і аудіо-записами цих книг. Створіть клас publication, в якому зберігаються назва (рядок) і ціна (тип float) книги. Від цього класу успадковуються ще два класи: book, який містить інформацію про кількість сторінок у книзі (типу int), і type, який містить час запису книги у хвилинах (тип float).
У кожному з цих трьох класів повинен бути метод getdata(), через який можна отримувати дані від користувача з клавіатури, і putdata(), призначений для виведення цих даних.
Напишіть функцію main() програми для перевірки класів book і type. Створіть їх об'єкти в програмі і запросіть користувача ввести і вивести дані з використанням методів getdataQ і putdata().

Код програми:

Publication.h:

```
#pragma once
class Publication {
private:
	char* name;
	int price;
public:
	Publication();
	Publication(char*, int);
	Publication(const Publication&);
	~Publication();
	void setName(char*);
	void setPrice(int);
	char* getName() const;
	int getPrice() const;
	void setData();
	void getData();
};
```

Publication.cpp:

```
#include "Publication.h"
#include <string.h>
#include <iostream>

Publication::Publication() : name(nullptr), price(0) {}
Publication::Publication(char* nName, int nPrice){
	setName(nName);
	setPrice(nPrice);
}
Publication::Publication(const Publication& old){
```

```
	setName(old.name);
	setPrice(old.price);
}

Publication::~Publication() {
	if (this->name) {
		delete[] this->name;
	}
}

void Publication::setName(char* nName) {
	if (this->name) {
		delete[] this->name;
	}
	int len = strlen(nName) + 1;
	this->name = new char[len];
	strcpy_s(this->name, len, nName);
}
void Publication::setPrice(int nPrice) {
	this->price = nPrice;
}

char* Publication::getName() const {
	return this->name;
}
int Publication::getPrice() const {
	return this->price;
}

void Publication::setData() {
	std::cout << "Enter name of the publication: ";
	char tmpname[64];
	int tmpprice;
	std::cin.getline(tmpname, 64);
	setName(tmpname);
	std::cout << "Enter price: ";
	std::cin >> tmpprice;
	setPrice(tmpprice);
}

void Publication::getData() {
	std::cout << "Name is \"" << this->getName() << "\"" << std::endl;
	std::cout << "Price is " << this->getPrice() << std::endl;
}
```

Book.h:

```
#pragma once
#include "Publication.h"

class Book : public Publication {
private:
	int pages;
public:
	Book();
	void setPages(int);
	int getPages() const;
	void getData() const;
	void setData();
};
```

Book.cpp:

```cpp
#include "Book.h"
#include <iostream>

Book::Book() : Publication(), pages(0) {}
void Book::setData() {
	int tmppages;
	Publication::setData();
	std::cout << "Enter pages in the book: ";
	std::cin >> tmppages;
	setPages(tmppages);
}

void Book::getData() const {
	std::cout << "Name is \"" << this->getName() << "\"" << std::endl;
	std::cout << "Price is " << this->getPrice() << std::endl;
	std::cout << "Price is " << getPages() << std::endl;
}

int Book::getPages() const {
	return this->pages;
}

void Book::setPages(int p) {
	this->pages = p;
}
```

Tape.h:

```cpp
#pragma once
#include "Publication.h"

class Tape : public Publication {
private:
	int audio_duration;
public:
	Tape();
	void setADur(int);
	int getADur() const;
	void getData() const;
	void setData();
};
```

Tape.cpp:

```cpp
#include "Tape.h"
#include <iostream>

Tape::Tape() : Publication(), audio_duration(0) {}
void Tape::setData() {
	int tmpdur;
	Publication::setData();
	std::cout << "Enter audio duration of the tape: ";
	std::cin >> tmpdur;
	setADur(tmpdur);
}

void Tape::getData() const {
	std::cout << "Name is \"" << this->getName() << "\"" << std::endl;
	std::cout << "Price is " << this->getPrice() << std::endl;
	std::cout << "Price is " << getADur() << std::endl;}
```

```cpp
int Tape::getADur() const {
	return this->audio_duration;
}

void Tape::setADur(int p) {
	this->audio_duration = p;
}
```

task1.cpp:

```cpp
#include <iostream>
#include "Book.h"
#include "Tape.h"
using namespace std;

int main()
{
    Book b;
    b.setData();
    b.getData();
    Tape t;
    cin.ignore();
    t.setData();
    t.getData();
    return 0;
}
```

Результат:

```
Enter name of the publication: кобзар
Enter price: 150
Enter pages in the book: 300
Name is "кобзар"
Price is 150
Price is 300
Enter name of the publication: роботи та машини
Enter price: 400
Enter audio duration of the tape: 58
Name is "роботи та машини"
Price is 400
Price is 58
```

Завдання 2. До класів з попереднього завдання (попередньо зберігши окремо код) додайте базовий клас sales, в якому міститься масив, що складається з трьох значень типу float, куди можна записати загальну вартість проданих книг за останні три місяці.
Включіть в клас методи getdata() для отримання значень вартості від користувача і putdata() для виведення цих цифр. Змініть класи book і type так, щоб вони стали похідними обох класів: publication і sales.
Об'єкти класів book і type повинні вводити і виводити дані про продажі разом з іншими своїми даними.
Напишіть функцію main() для створення об'єктів класів book і type, щоб протестувати можливості введення/виведення даних.

Код програми:

Publication.h:

```
#pragma once
class Publication {
private:
	char* name;
	int price;
public:
	Publication();
	Publication(char*, int);
	Publication(const Publication&);
	~Publication();
	void setName(char*);
	void setPrice(int);
	char* getName() const;
	int getPrice() const;
	void setData();
	void getData();
};
```

Publication.cpp:

```
#include "Publication.h"
#include <string.h>
#include <iostream>

Publication::Publication() : name(nullptr), price(0) {}
Publication::Publication(char* nName, int nPrice) {
	setName(nName);
	setPrice(nPrice);
}
Publication::Publication(const Publication& old) {
	setName(old.name);
	setPrice(old.price);
}

Publication::~Publication() {
	if (this->name) {
		delete[] this->name;
	}
}

void Publication::setName(char* nName) {
	if (this->name) {
		delete[] this->name;
	}
	int len = strlen(nName) + 1;
	this->name = new char[len];
	strcpy_s(this->name, len, nName);
}
void Publication::setPrice(int nPrice) {
	this->price = nPrice;
}

char* Publication::getName() const {
	return this->name;
}
int Publication::getPrice() const {
	return this->price;
}
```

```
void Publication::setData() {
	std::cout << "Enter name of the publication: ";
	char tmpname[64];
	int tmpprice;
	std::cin.getline(tmpname, 64);
	setName(tmpname);
	std::cout << "Enter price: ";
	std::cin >> tmpprice;
	setPrice(tmpprice);
}

void Publication::getData() {
	std::cout << "Name is \"" << this->getName() << "\"" << std::endl;
	std::cout << "Price is " << this->getPrice() << std::endl;
}
```

Sales.h:

```
#pragma once
class Sales {
private:
	float totalPrice[3];
public:
	Sales();
	void setData();
	void getData() const;
};
```

Sales.cpp:

```
#include <iostream>
#include "Sales.h"

Sales::Sales() {
	for (int i = 0; i < 3; i++)
		this->totalPrice[i] = 0;
}
void Sales::setData() {
	for (int i = 0; i < 3; i++) {
		std::cout << "Enter sales for " << i << " mounth: ";
		std::cin >> this->totalPrice[i];
	}
}
void Sales::getData() const {
	for (int i = 0; i < 3; i++) {
		std::cout << "Sales for " << i << " mounth are: " << this->totalPrice[i]
<< std::endl;
	}
}
```

Book.h:

```
#pragma once
#include "Publication.h"
#include "Sales.h"

class Book : public Publication, public Sales {
private:
	int pages;
public:
```

```
	Book();
	void setPages(int);
	int getPages() const;
	void getData() const;
	void setData();
};
```

Book.cpp:

```
#include "Book.h"
#include "Sales.h"
#include <iostream>

Book::Book() : Publication(), pages(0) {}
void Book::setData() {
	int tmppages;
	Publication::setData();
	Sales::setData();
	std::cout << "Enter pages in the book: ";
	std::cin >> tmppages;
	setPages(tmppages);}

void Book::getData() const {
	std::cout << "Name is \"" << this->getName() << "\"" << std::endl;
	std::cout << "Price is " << this->getPrice() << std::endl;
	std::cout << "Pages in the book: " << getPages() << std::endl;
	Sales::getData();}

int Book::getPages() const {
	return this->pages;}

void Book::setPages(int p) {
	this->pages = p;}
```

Tape.h:

```
#pragma once
#include "Publication.h"
#include "Sales.h"

class Tape : public Publication, public Sales {
private:
	int audio_duration;
public:
	Tape();
	void setADur(int);
	int getADur() const;
	void getData() const;
	void setData();
};
```

Tape.cpp:

```
#include "Tape.h"
#include "Sales.h"
#include <iostream>

Tape::Tape() : Publication(), audio_duration(0) {}
void Tape::setData() {
	int tmpdur;
	Publication::setData();
```

```
	Sales::setData();
	std::cout << "Enter audio duration of the tape: ";
	std::cin >> tmpdur;
	setADur(tmpdur);}

void Tape::getData() const {
	std::cout << "Name is \"" << this->getName() << "\"" << std::endl;
	std::cout << "Price is " << this->getPrice() << std::endl;
	std::cout << "Audio duration is " << getADur() << std::endl;
	Sales::getData();}

int Tape::getADur() const {
	return this->audio_duration;}

void Tape::setADur(int p) {
	this->audio_duration = p;}
```

task2.cpp:

```
#include <iostream>
#include "Book.h"
#include "Tape.h"
using namespace std;

int main(){
    Book b;
    b.setData();
    b.getData();
    Tape t;
    cin.ignore();
    t.setData();
    t.getData();
    return 0;
}
```

Результат:

```
Enter name of the publication: книжка письмова
Enter price: 300
Enter sales for 1 mounth: 600
Enter sales for 2 mounth: 1200
Enter sales for 3 mounth: 1800
Enter pages in the book: 256
Name is "книжка письмова"
Price is 300
Pages in the book: 256
Sales for 1 mounth are: 600
Sales for 2 mounth are: 1200
Sales for 3 mounth are: 1800
Enter name of the publication: книжка не письмова
Enter price: 150
Enter sales for 1 mounth: 450
Enter sales for 2 mounth: 300
Enter sales for 3 mounth: 600
Enter audio duration of the tape: 21
Name is "книжка не письмова"
Price is 150
Audio duration is 21
Sales for 1 mounth are: 450
Sales for 2 mounth are: 300
Sales for 3 mounth are: 600
```

Індивідуальне завдання: варіант 10.
Створити клас ЧОТИРИКУТНИК з полями – координатами вершин. Визначити конструктори, деструктор, функції доступу до полів, введення-виведення та обчислення добутку чисел.
Створити похідний клас ПАРАЛЕЛОГРАМ. Визначити конструктори за замовчуванням і з різним числом параметрів, деструктор, функції доступу до полів, введення-виведення, обчислення площі паралелограма.

Код програми:

Quadrangle.h:

```
#pragma once

class Quadrangle {
private:
	//float upperLeft[2];                                                    // index 0
is x, index 1 is y
	//float upperRight[2];
	//float lowerLeft[2];
	//float lowerRight[2];

	float upperLeft_x;
	float upperLeft_y;
	float upperRight_x;
	float upperRight_y;
	float lowerLeft_x;
	float lowerLeft_y;
	float lowerRight_x;
	float lowerRight_y;

public:
	Quadrangle();

	Quadrangle(float x1, float y1, float x2, float y2, float x3, float y3, float x4,
float y4);
	void setUpperLeft(float x, float y);

	void setUpperRight(float x, float y);
	void setLowerLeft(float x, float y);

	void setLowerRight(float x, float y);

	float getUpperLeft_x();

	float getUpperLeft_y();



	float getUpperRight_x();

	float getUpperRight_y();



	float getLowerLeft_x();

	float getLowerLeft_y();
```

```
	float getLowerRight_x();

	float getLowerRight_y();
	float mult();
	~Quadrangle();
};
```

Quadrangle.cpp:

```
#include "Quadrangle.h"

Quadrangle::Quadrangle() {
	setUpperLeft(0, 0);
	setUpperRight(0, 0);
	setLowerLeft(0, 0);
	setLowerRight(0, 0);
}

Quadrangle::Quadrangle(float x1, float y1, float x2, float y2, float x3, float y3,
float x4, float y4) {
	setUpperLeft(x1, y1);
	setUpperRight(x2, y2);
	setLowerLeft(x3, y3);
	setLowerRight(x4, y4);
}

void Quadrangle::setUpperLeft(float x, float y) {
	upperLeft_x = x;
	upperLeft_y = y;
}

void Quadrangle::setUpperRight(float x, float y) {
	upperRight_x = x;
	upperRight_y = y;
}

void Quadrangle::setLowerLeft(float x, float y) {
	lowerLeft_x = x;
	lowerLeft_y = y;
}

void Quadrangle::setLowerRight(float x, float y) {
	lowerRight_x = x;
	lowerRight_y = y;
}

float Quadrangle::getUpperLeft_x() {
	return this->upperLeft_x;
}

float Quadrangle::getUpperLeft_y() {
	return this->upperLeft_y;
}



float Quadrangle::getUpperRight_x() {
	return this->upperRight_x;
}

float Quadrangle::getUpperRight_y() {
	return this->upperRight_y;
}
```

```
float Quadrangle::getLowerLeft_x() {
	return this->lowerLeft_x;
}

float Quadrangle::getLowerLeft_y() {
	return this->lowerLeft_y;
}



float Quadrangle::getLowerRight_x() {
	return this->lowerRight_x;
}

float Quadrangle::getLowerRight_y() {
	return this->lowerRight_y;
}
float Quadrangle::mult() {
	return upperLeft_x * upperLeft_y * upperRight_x * upperRight_y *
		lowerLeft_x * lowerLeft_y * lowerRight_x * lowerRight_y;
}
Quadrangle::~Quadrangle() {}
```

Parallelogram.h:

```
#pragma once
#include "Quadrangle.h"
#include <cmath>
#include <iostream>
class Parallelogram : public Quadrangle {
private:
	float sideA;
	float sideB;
	float angle;
public:
	Parallelogram();
	Parallelogram(float, float, float, float, float, float, float, float); //bad
	~Parallelogram();
	void setSideA(float);
	void setSideB(float);
	void setAngle(float);

	float getSideA(float);
	float getSideB(float);
	float getAngle(float);

	float square();

	void setParall();
};
```

Parallelogram.cpp:

```
#include "Parallelogram.h"

Parallelogram::Parallelogram() : Quadrangle() {
	this->sideA = 0;
	this->sideB = 0;
	this->angle = 0;
}
```

```
Parallelogram::Parallelogram(float x1, float y1, float x2, float y2, float x3, float
y3, float x4, float y4) :
Quadrangle(x1, y1, x2, y2, x3, y3, x4, y4) {}

Parallelogram::~Parallelogram() {}

float Parallelogram::square() {
	return this->sideA * this->sideB * sin(this->angle * 3.14159 / 180);
}

void Parallelogram::setSideA(float a) {
	this->sideA = a;
}
void Parallelogram::setSideB(float a) {
	this->sideB = a;
}
void Parallelogram::setAngle(float a) {
	this->angle = a;
}

float Parallelogram::getSideA(float a) {
	return this->sideA;
}
float Parallelogram::getSideB(float a) {
	return this->sideB;
}
float Parallelogram::getAngle(float a) {
	return this->angle;
}

void Parallelogram::setParall() {
	float temp;
	std::cout << "Enter side a: ";
	std::cin >> temp;
	setSideA(temp);
	std::cout << "Enter side b: ";
	std::cin >> temp;
	setSideB(temp);
	std::cout << "Enter angle: ";
	std::cin >> temp;
	setAngle(temp);
}
```

individual.cpp:

```
#include <iostream>
#include "Parallelogram.h"
#include "Quadrangle.h"
using namespace std;
int main()
{
    Quadrangle qr;
    Parallelogram par(-1, 3, 2, 3, -2, 1, 1, 1);
    cout << par.mult() << endl;
    par.setParall();
    cout << "Square is " << par.square() << endl;
    return 0;
}
```

Результат:

```
36
Enter side a: 10
Enter side b: 5
Enter angle: 30
Square is 25
```

Висновок: ознайомився зі способами та механізмами успадкування класів та навчився використовувати їх для побудови об’єктно-орієнтованих програм.