Лабораторна робота №5

Тема: Віртуальні функції та поліморфізм.
Мета: Практично ознайомитись з поняттям поліморфізму, його застосуванням та вивчити механізм його реалізації за допомогою віртуальних функцій

Завдання 1. Нехай є видавнича компанія, яка описана в завданні 1 попередньої лабораторної роботи, яка продає і книги, і аудіо версії друкованої продукції. Як і в тому завданні, створіть клас publication, який зберігає назву (фактично, рядок) і ціну (тип float) публікації.
Створіть два похідних класа: book, який містить інформацію про кількість сторінок у книзі (типу int), і type, який містить час запису аудіокниги у хвилинах (тип float). Кожен з класів повинен мати віртуальний метод getdata(), який буде запитувати інформацію у користувача, і віртуальний метод putdata() для виведення даних на екран.
Напишіть функцію main(), в якій створіть масив вказівників на клас publication: publication* arr[4];
У циклі while() запитуйте у користувача, який об’єкт потрібно створити (використовуйте new для створення нового об'єкта book або tape). Після чого за допомогою метод getdata() в атрибути об’єктів вносити дані відповідно до типу об’єкта.
Коли користувач закінчить введення вихідних даних, виведіть результат для всіх введених книг і касет, використовуючи цикл for і єдиний вираз: arr[i]->putdata(); для виведення даних про кожен об'єкт з масиву.

Код програми:

Publication.h:

```
#pragma once

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

Publication.cpp:

```
#include "Publication.h"
```

```cpp
#include <string.h>
#include <string>
#include <iostream>
using namespace std;

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

Book.h:

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

class Book : public Publication {
private:
```

```
	int pages;
public:
	Book();
	Book(const Book&);
	void setPages(int);
	int getPages() const;
	void getData() const override;
	void setData() override;
};
```

Book.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 << "Pages is " << getPages() << std::endl;
}

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

void Book::setPages(int p) {
	this->pages = p;
}
Book::Book(const Book& old) {
	setPages(old.getPages());
	setName(old.getName());
	setPrice(old.getPrice());
}
```

Tape.h:

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

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

Tape.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 << "Audio duration is " << getADur() << std::endl;
}

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

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

Tape::Tape(const Tape& old) {
	setADur(old.getADur());
	setName(old.getName());
	setPrice(old.getPrice());
}
```

task1.cpp:

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

int main()
{
    short guess;
    Publication *arr[4];

    for (int i = 0; i < 4; i++) {
        do {
            cout << "Create publication Book (1), Tape (2): ";
            cin >> guess;
            cin.ignore();
            switch (guess) {
            case 1:
                cout << "You chose Book" << endl;
                arr[i] = new Book();
                arr[i]->setData();
                break;
            case 2:
                cout << "You chose Tape" << endl;
                arr[i] = new Tape();
                arr[i]->setData();
```

```
                break;
            default:
                cout << "Wrong category. Try again" << endl;

                break;
            }
        } while (guess != 1 && guess != 2);
    }

    for (int i = 0; i < 4; i++) {
        arr[i]->getData();
    }

    return 0;
}
```

Результат:

```
Create publication Book (1), Tape (2): 1
You chose Book
Enter name of the publication: Theory of algorithms
Enter price: 500
Enter pages in the book: 450
Create publication Book (1), Tape (2): 2
You chose Tape
Enter name of the publication: Ghost in the shell
Enter price: 210
Enter audio duration of the tape: 126
Create publication Book (1), Tape (2): 1
You chose Book
Enter name of the publication: Sockets
Enter price: 240
Enter pages in the book: 150
Create publication Book (1), Tape (2): 2
You chose Tape
Enter name of the publication: Motivate yourself now!
Enter price: 70
Enter audio duration of the tape: 120
Name is "Theory of algorithms"
Price is 500
Pages is 450
Name is "Ghost in the shell"
Price is 210
Audio duration is 126
Name is "Sockets"
Price is 240
Pages is 150
Name is "Motivate yourself now!"
Price is 70
Audio duration is 120
```

Завдання 2. Взявши за основу програму із завдання 1, додайте до класів book і tape метод isOveersize (), який повертає значення типу bool.
Припустимо, книга, в якій більше 800 сторінок, або аудіо запис, з часом програвання якого більше 90 хвилин, будуть вважатися об'єктами з перевищенням розміру.
До цієї функції можна звертатися з main(), а результат її роботи виводити у вигляді рядка «Перевищення розміру!» для відповідних книг і касет. Об'єкти класів book і tape повинні зберігаються в масиві типу publication*.

Publication.h:

```
#pragma once

class Publication {
private:
	char* name;
	int price;
public:
	Publication();
	~Publication();
	Publication(char*, int);
	Publication(const Publication&);
	void setName(char*);
	void setPrice(int);
	char* getName() const;
	int getPrice() const;
	virtual void setData();
	virtual void getData() const;
	virtual bool isOversize() const = 0 {};
};
```

Publication.cpp:

```
#include "Publication.h"
#include <string.h>
#include <string>
#include <iostream>
using namespace std;

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() const {
	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();
	Book(const Book&);
	void setPages(int);
	int getPages() const;
	void getData() const override;
	void setData() override;
	bool isOversize() const;
};
```

Book.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 << "Pages is " << getPages() << std::endl;
}

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

void Book::setPages(int p) {
	this->pages = p;
}
Book::Book(const Book& old) {
	setPages(old.getPages());
	setName(old.getName());
	setPrice(old.getPrice());
}

bool Book::isOversize() const {
	if (this->pages > 800) {
		return true;
	}
	return false;
}
```

Tape.h:

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

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

Tape.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 << "Audio duration is " << getADur() << std::endl;
```

```
}

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

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

Tape::Tape(const Tape& old) {
	setADur(old.getADur());
	setName(old.getName());
	setPrice(old.getPrice());
}

bool Tape::isOversize() const {
	if (this->audio_duration > 90) {
		return true;
	}
	return false;
}
```

task2.cpp:

```
#include <iostream>
#include <conio.h>
#include "Publication.h"
#include "Book.h"
#include "Tape.h"

#define N 2

using namespace std;

int main()
{
    short guess;
    Publication* arr[4];

    for (int i = 0; i < N; i++) {
        do {
            cout << "Create publication Book (1), Tape (2): ";
            cin >> guess;
            cin.ignore();
            switch (guess) {
            case 1:
                cout << "You chose Book" << endl;
                arr[i] = new Book();
                arr[i]->setData();
                break;
            case 2:
                cout << "You chose Tape" << endl;
                arr[i] = new Tape();
                arr[i]->setData();
                break;
            default:
                cout << "Wrong category. Try again" << endl;

                break;
            }
        } while (guess != 1 && guess != 2);
    }
```

```
    for (int i = 0; i < N; i++) {
        arr[i]->getData();
        if (arr[i]->isOversize()) {
            cout << "Oversized!" << endl;
        }
    }

    return 0;
}
```

Результат:

```
Create publication Book (1), Tape (2): 1
You chose Book
Enter name of the publication: some book
Enter price: 550
Enter pages in the book: 860
Create publication Book (1), Tape (2): 2
You chose Tape
Enter name of the publication: audio tape
Enter price: 300
Enter audio duration of the tape: 40
Name is "some book"
Price is 550
Pages is 860
Oversized!
Name is "audio tape"
Price is 300
Audio duration is 40
```

Індивідуальне завдання:
Варіант 10. Створити клас ЧОТИРИКУТНИК з полями – довжинами сторін. Визначити віртуальну функцію обчислення периметра.
Створити похідні класи ПАРАЛЕЛОГРАМ, РОМБ зі своїми функціями обчислення периметра.
Для перевірки використати масив вказівників на об’єкти базового класу, яким присвоїти адреси об’єктів похідних класів.

Quadrangle.h:

```
#pragma once

class Quadrangle {
private:

      //     abbc
      //    a   c
      //   addddc

      float sideA;
      float sideB;
      float sideC;
      float sideD;

public:
      Quadrangle();
```

```
        Quadrangle(float, float, float, float);

        void setSideA(float);
        float getSideA() const;
        void setSideB(float);
        float getSideB() const;
        void setSideC(float);
        float getSideC() const;
        void setSideD(float);
        float getSideD() const;
        virtual float perimeter() const;
        ~Quadrangle();
};
```

Quadrangle.cpp:

```
#include "Quadrangle.h"

Quadrangle::Quadrangle() : sideA(0), sideB(0), sideC(0), sideD(0) {}

Quadrangle::Quadrangle(float a, float b, float c, float d) : sideA(a), sideB(b),
sideC(c), sideD(d) {}


void Quadrangle::setSideA(float a) {
        this->sideA = a;
}

void Quadrangle::setSideB(float b) {
        this->sideB = b;
}

float Quadrangle::getSideA() const {
        return this->sideA;
}

float Quadrangle::getSideB() const {
        return this->sideB;
}



void Quadrangle::setSideC(float c) {
        this->sideC = c;
}

void Quadrangle::setSideD(float d) {
        this->sideD = d;
}

float Quadrangle::getSideC() const {
        return this->sideC;
}

float Quadrangle::getSideD() const {
        return this->sideD;
}
float Quadrangle::perimeter() const {
        return sideA + sideB + sideC + sideD;
}
Quadrangle::~Quadrangle() {}
```

Parallelogram.h:

```
#pragma once
#include "Quadrangle.h"
#include <cmath>
#include <iostream>
class Parallelogram : public Quadrangle {
private:
	//      abba
	//     a  a
	//    abba

public:
	Parallelogram();
	Parallelogram(float, float);
	~Parallelogram();
	float perimeter() const override;
};
```

Parallelogram.cpp:

```
#include "Parallelogram.h"
#include "Quadrangle.h"

Parallelogram::Parallelogram() : Quadrangle() {};

Parallelogram::Parallelogram(float a, float b) :
	Quadrangle(a, b, a, b) {}


Parallelogram::~Parallelogram() {}

float Parallelogram::perimeter() const {
	return 2 * (getSideA() + getSideB());
}
```

Rhombus.h:

```
#pragma once
#include "Quadrangle.h"
#include "Parallelogram.h"

class Rhombus : public Parallelogram {
private:

	//      abbba
	//     a   a
	//    abbba
public:
	Rhombus();
	Rhombus(float);
	~Rhombus();

	float perimeter() const override;
};
```

Rhombus.cpp:

```
#include "Rhombus.h"
#include "Quadrangle.h"
```

```
Rhombus::Rhombus() : Parallelogram() {}
Rhombus::Rhombus(float a) : Parallelogram(a, a) {}
Rhombus::~Rhombus(){}
float Rhombus::perimeter() const  {
      return 4 * getSideA();
}
```

individual.cpp:

```
#include <iostream>
#include "Parallelogram.h"
#include "Quadrangle.h"
#include "Rhombus.h"

#define N 4

using namespace std;
int main()
{
    short guess;
    Quadrangle* arr[4];

    for (int i = 0; i < N; i++) {
        do {
            cout << "Create figure: Quadrangle (1), Parallelogram (2), Rhombus (3): ";
            cin >> guess;
            cin.ignore();
            switch (guess) {
            case 1:
                cout << "You chose Quadrangle" << endl;
                arr[i] = new Quadrangle(2, 3, 4, 5);
                break;
            case 2:
                cout << "You chose Parallelogram" << endl;
                arr[i] = new Parallelogram(4, 5);
                break;
            case 3:
                cout << "You chose Rhombus" << endl;
                arr[i] = new Rhombus(2);
                break;
            default:
                cout << "Wrong category. Try again" << endl;

                break;
            }
        } while (guess < 1 || guess >= N);
    }

    for (int i = 0; i < N; i++) {
        cout << arr[i]->perimeter() << endl;
        if (arr[i]) {
            delete arr[i];
        }
    }
    return 0;
}
```

Результат:

```
Create figure: Quadrangle (1), Parallelogram (2), Rhombus (3): 1
You chose Quadrangle
Create figure: Quadrangle (1), Parallelogram (2), Rhombus (3): 2
You chose Parallelogram
Create figure: Quadrangle (1), Parallelogram (2), Rhombus (3): 3
You chose Rhombus
Create figure: Quadrangle (1), Parallelogram (2), Rhombus (3): 4
Wrong category. Try again
Create figure: Quadrangle (1), Parallelogram (2), Rhombus (3): 1
You chose Quadrangle
14
18
8
14
```

Висновок: практично ознайомитисья з поняттям поліморфізму, його застосуванням та вивчив механізм його реалізації за допомогою віртуальних функцій.