Міністерство освіти і науки України

Відокремлений структурний підрозділ «Тернопільський фаховий коледж Тернопільського національного технічного університету імені Івана Пулюя»

Циклова комісія комп'ютерних наук

ЗВІТ

про виконання лабораторної роботи №5

з дисципліни: "Об'єктно-орієнтоване програмування"

на тему: "Поліморфізм та віртуальні методи"

Виконала:
студентка групи КН-321
Байдецька. В. П.

Прийняв:
Слободян Р.О

Тернопіль 2025

Мета: Практично ознайомитись з поняттям поліморфізму, його застосуванням та вивчити механізм його реалізації за допомогою віртуальних методів

Хід роботи

Поліморфізм є однією з трьох основних особливостей ООП. У мові C++ це механізм реалізації функції, у якому різні результати можна отримати за допомогою одного її імені. Поліморфізм характеризується фразою "один інтерфейс, багато методів", що означає доступ до всіх функцій-членів класу одним способом, незважаючи на відмінності в їх конкретних діях.

У C++ поліморфізм підтримується як під час компіляції, так і під час виконання програми. Перевантаження операторів і функцій є прикладом статичного поліморфізму, що реалізується в момент компілювання. Однак цей механізм не здатний вирішити всі завдання реальних додатків з ООП. Тому в C++ також реалізовано динамічний поліморфізм періоду виконання, який базується на використанні похідних класів через вказівники на базовий клас і віртуальних функцій.

Віртуальна функція оголошується в базовому класі з ключовим словом virtual і перевизначається в похідних класах. Кожен похідний клас може мати власну версію віртуальної функції, яку можна позначати ключовим словом override. При виклику віртуальної функції через вказівник або посилання на базовий клас програма визначає потрібну версію за типом об'єкта, а не за типом вказівника. Це рішення приймається під час виконання програми, тому при вказанні на різні об'єкти викликатимуться різні версії віртуальної функції.

Функція оголошується віртуальною в базовому класі за допомогою ключового слова virtual. При перевизначенні віртуальної функції у похідному класі ключове слово virtual повторювати не потрібно, хоча це не буде помилкою.

Завдання 1. Нехай є видавнича компанія, яка описана в завданні 1 попередньої лабораторної роботи, яка продає і книги, і аудіо версії книг. Як і в тому завданні, створіть клас Publication, який має атрибути: назву (тип string) і ціну (тип float) публікації. Створіть два похідних класи: Book, в якому є атрибут, що зберігає кількість сторінок (тип int), і AudioBook, в якому є атрибут, що зберігає тривалість аудіо книги в хвилинах (тип float). Всі класи повинні мати визначені віртуальні методи getData(), який буде запитувати інформацію у користувача, і printData(), який буде виводити дані на екран. Напишіть функцію main(), в якій створіть масив publications розміром не менше 4 елементи, який міститиме вказівники на об’єкти класу Publication (тип Publication*). За допомогою циклу for заповніть елементи масиву – у циклі запитуйте у користувача, що потрібно створити: книгу чи аудіокнигу (використовуйте функцію new для створення нового об'єкта класу Book або AudioBook), після чого потрібно ввести дані книги чи аудіокниги за допомогою виклику метода getData(). Коли користувач закінчить введення даних, потрібно вивести результат для всіх введених книг і аудіокниг, використовуючи цикл for і єдиний вираз у ньому: publications[i]–>printData()

Код:

```
#include <iostream>
#include <string>
using namespace std;
#define n 5
class Publication{
	string title;
	float price;
public:
	Publication();
	Publication* operator[](const int i){
		Publication* pointer;
		pointer = this;
		return pointer;
	}
	virtual void InputData();
	virtual void OutputData() const;
	virtual ~Publication(){
		cout << "object destroyed"<<endl;
	}
```

```
};
      Publication::Publication(){
            title = ' ';
            price = 0.0;
      }
      void Publication::InputData(){
            cout<<"BOOK"<<endl;
            cout<<"please enter title of the book ";
            cin >> title;
            cout<<"please enter price of the book ";
            cin >> price;
      }

      void Publication::OutputData() const{
            cout<<"book title: "<<title<<endl<<"book price: "<<price<<endl;
      }


class Book:public Publication {
      int pageCount;
public:
      Book();
      void InputData() override;
      void OutputData() const override;

};

      Book::Book(){
            pageCount = 0;
      }

      void Book::InputData(){
            Publication::InputData();
            cout<<"please enter number of pages in the book ";
            cin >> pageCount;
      }

      void Book::OutputData() const{
            Publication::OutputData();
            cout<<"there are "<<pageCount<<" pages in the book"<<endl;
      }

class AudioBook:public Publication{
      float duration;
public:
      AudioBook();
      void InputData() override;
      void OutputData() const override;



};
      AudioBook::AudioBook(){
            duration = 0.0;
      }

      void AudioBook::InputData() {
            cout<<"AUDIOBOOK"<<endl;
```

```
            Publication::InputData();
            cout<<"please enter duration of audiobook ";
            cin >> duration;

      }

      void AudioBook::OutputData() const{
            Publication::OutputData();
            cout<<"audio is "<<duration<<"minutes long"<<endl;
      }
int main()
{
      //ввід
      Publication *publications[n];
      for (int i = 0; i < n; ++i)
      {
            int choice;
            Publication publication;
            Publication* pointerToPublication;
            cout<<"Press 1 for book and 2 for audiobook ";
            cin>>choice;
            //вибрати аудіо чи паперова книига
            switch(choice){
                  case 1:
                        {
                              Book* pointerToBook = new Book();
                              pointerToBook->InputData();
                              pointerToPublication = pointerToBook;
                              break;
                        }
                  case 2:
                        {
                              AudioBook* pointerToAudioBook = new AudioBook();
                              pointerToAudioBook->InputData();
                              pointerToPublication = pointerToAudioBook;
                              break;
                        }
            }


            publications[i] = pointerToPublication;
      }
//вивід ввід
      for (int i = 0; i < n; ++i)
      {
            publications[i]->OutputData();
      }
      //звільнення динамічної пам'яті
      for (int i = 0; i < n; ++i)
      {
            delete publications[i];
      }
      return 0;
      }
```

```
please enter price of the book 1
please enter number of pages in the book 900
object destroyed
Press 1 for book and 2 for audiobook 2
AUDIOBOOK
BOOK
please enter title of the book b
please enter price of the book 4
please enter duration of audiobook 90
object destroyed
Press 1 for book and 2 for audiobook 1
BOOK
please enter title of the book c
please enter price of the book 90
please enter number of pages in the book 500
object destroyed
Press 1 for book and 2 for audiobook 2
AUDIOBOOK
BOOK
please enter title of the book d
please enter price of the book 5
please enter duration of audiobook 120
object destroyed
Press 1 for book and 2 for audiobook 1
BOOK
please enter title of the book e
please enter price of the book 99
please enter number of pages in the book 100
object destroyed
book title: a
book price: 1
there are 900 pages in the book
book title: b
book price: 4
audio is 90minutes long
book title: c
book price: 90
there are 500 pages in the book
book title: d
book price: 5
audio is 120minutes long
book title: e
book price: 99
there are 100 pages in the book
object destroyed
object destroyed
object destroyed
object destroyed
object destroyed
[cheshyrka@archlinux OOP]$
```

Завдання 2. Взявши за основу програму із завдання 1, додайте віртуальний метод який називається isOversized(), до класів Book і AudioBook. Припустимо, книга, в якій більше 800 сторінок, або аудіокнига з тривалістю більшою за 90 хвилин, будуть вважатися об'єктами з перевищенням розміру, тоді метод isOversized () повертатиме true. Перевірте розмір/тривалість всіх публікацій, які знаходяться у масиві за допомогою виклику нового методу, якщо буде перевищення вказаних лімітів, потрібно виводити рядок «Перевищення розміру!» і назву публікації

Код:

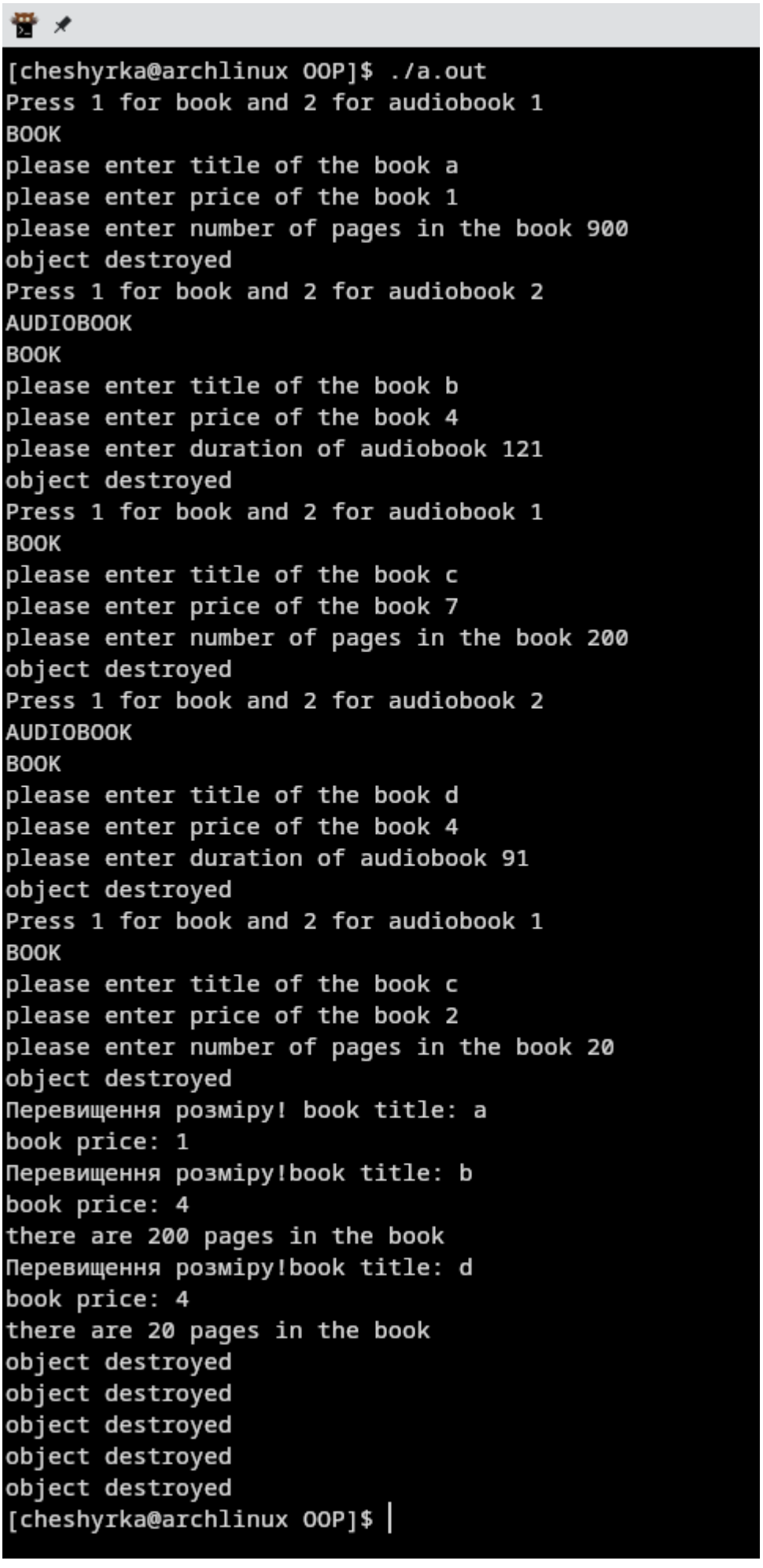
#include <iostream>

```
#include <string>
using namespace std;
#define n 5
class Publication{
	string title;
	float price;
public:
	Publication();
	Publication* operator[](const int i){
		Publication* pointer;
		pointer = this;
		return pointer;
	}
	virtual void InputData();
	virtual void OutputData() const;
	virtual bool isOversized() const;
	virtual ~Publication(){
		cout << "object destroyed"<<endl;
	}

};
	Publication::Publication(){
		title = ' ';
		price = 0.0;
	}
	bool Publication::isOversized() const{
		return false;
	}
	void Publication::InputData(){
		cout<<"BOOK"<<endl;
		cout<<"please enter title of the book ";
		cin >> title;
		cout<<"please enter price of the book ";
		cin >> price;
	}

	void Publication::OutputData() const{
		cout<<"book title: "<<title<<endl<<"book price: "<<price<<endl;
```

```
	}


class Book:public Publication {
	int pageCount;
public:
	Book();
	void InputData();
	void OutputData() const;
	bool isOversized() const;
};

	Book::Book(){
		pageCount = 0;
	}

	bool Book::isOversized() const{
		if(this->pageCount >= 800)
			return true;
		else
			false;
	return false;
	}

	void Book::InputData(){
		Publication::InputData();
		cout<<"please enter number of pages in the book ";
		cin >> pageCount;
	}

	void Book::OutputData() const{
		if(this->isOversized()){
			cout<<"Перевищення розміру! ";
			Publication::OutputData();
		}
		else
			cout<<"there are "<<pageCount<<" pages in the book"<<endl;
	}

class AudioBook:public Publication{
	float duration;
public:
	AudioBook();
	void InputData();
	void OutputData() const;
	bool isOversized() const;

};
	AudioBook::AudioBook(){
		duration = 0.0;
	}

	bool AudioBook::isOversized() const{
		if(duration >= 90)
			return true;
		else
			return false;
```

```
	}

	void AudioBook::InputData() {
		cout<<"AUDIOBOOK"<<endl;
		Publication::InputData();
		cout<<"please enter duration of audiobook ";
		cin >> duration;

	}

	void AudioBook::OutputData() const{
		if(this->isOversized()){
			cout<<"Перевищення розміру!";
			Publication::OutputData();
		}
		else
			cout<<"audio is "<<duration<<"minutes long"<<endl;
	}
int main()
{
	//ввід
	Publication *publications[n];
	for (int i = 0; i < n; ++i)
	{
		int choice;
		Publication publication;
		Publication* pointerToPublication;
		cout<<"Press 1 for book and 2 for audiobook ";
		cin>>choice;
		//вибрати аудіо чи паперова книига
		switch(choice){
			case 1:
				{
					Book* pointerToBook = new Book();
					pointerToBook->InputData();
					pointerToPublication = pointerToBook;
					break;
				}
			case 2:
				{
					AudioBook* pointerToAudioBook = new AudioBook();
					pointerToAudioBook->InputData();
					pointerToPublication = pointerToAudioBook;
					break;
				}
		}


		publications[i] = pointerToPublication;
	}
//вивід
	for (int i = 0; i < n; ++i)
	{
		publications[i]->OutputData();
	}
```

```
	//звільнення динамічної пам'яті
	for (int i = 0; i < n; ++i)
	{
		delete publications[i];
	}
	return 0;
}
```
//done треба було тільки додати перевищення розміру


[cheshyrka@archlinux OOP]$ ./a.out
Press 1 for book and 2 for audiobook 1
BOOK
please enter title of the book a
please enter price of the book 1
please enter number of pages in the book 900
object destroyed
Press 1 for book and 2 for audiobook 2
AUDIOBOOK
BOOK
please enter title of the book b
please enter price of the book 4
please enter duration of audiobook 121
object destroyed
Press 1 for book and 2 for audiobook 1
BOOK
please enter title of the book c
please enter price of the book 7
please enter number of pages in the book 200
object destroyed
Press 1 for book and 2 for audiobook 2
AUDIOBOOK
BOOK
please enter title of the book d
please enter price of the book 4
please enter duration of audiobook 91
object destroyed
Press 1 for book and 2 for audiobook 1
BOOK
please enter title of the book c
please enter price of the book 2
please enter number of pages in the book 20
object destroyed
Перевищення розміру! book title: a
book price: 1
Перевищення розміру!book title: b
book price: 4
there are 200 pages in the book
Перевищення розміру!book title: d
book price: 4
there are 20 pages in the book
object destroyed
object destroyed
object destroyed
object destroyed
object destroyed
[cheshyrka@archlinux OOP]$

4*. Для заданого варіанта індивідуального завдання написати програму з

використанням динамічного поліморфізму. Для цього виконати наступне:

– Для похідних класів зробити віртуальними методи:

➢ input() – метод для запит у користувача даних та їх зчитування з клавіатури у поля класу;

➢ print() – константний метод виводу даних на екран. – В функції main створити масив розміром більше 5 елементів (тип елементів – вказівник на базовий клас згідно варіанту завдання). – Заповнити елементи масиву об’єктами похідних класів (вибір здійснює користувач за допомогою меню), заповнити дані об’єктів з використанням методу input(). – Вивести інформацію про об’єкти за допомогою циклу for з використанням методу print().

```
#include <iostream>
#include <cstring>
using namespace std;
#define n 5
//базовий клас
    class Toy {
        char *ownerName; // власник
        int old; // для якого віку іграшка
    public:
        Toy();
        Toy(char * ownerName, int old);
        Toy(const Toy& );
        void setOwnerName(char * ownerName);
        char * getOwnerName();
        void setOld(int old);
        int getOld();
        virtual void print() const;
        virtual void input();
        ~Toy();
    };

    // Реалізація методів класу
    Toy::Toy(){
        old = 0;
        ownerName = new char[1];
        ownerName[0] = '\0';
        cout << "object created " << endl;
    }

    Toy::Toy(char * ownerName, int old){
        this->old = old;
        this->ownerName = new char[strlen(ownerName) + 1];
        strcpy(this->ownerName, ownerName);
        cout << "object created " << endl;
    }
```

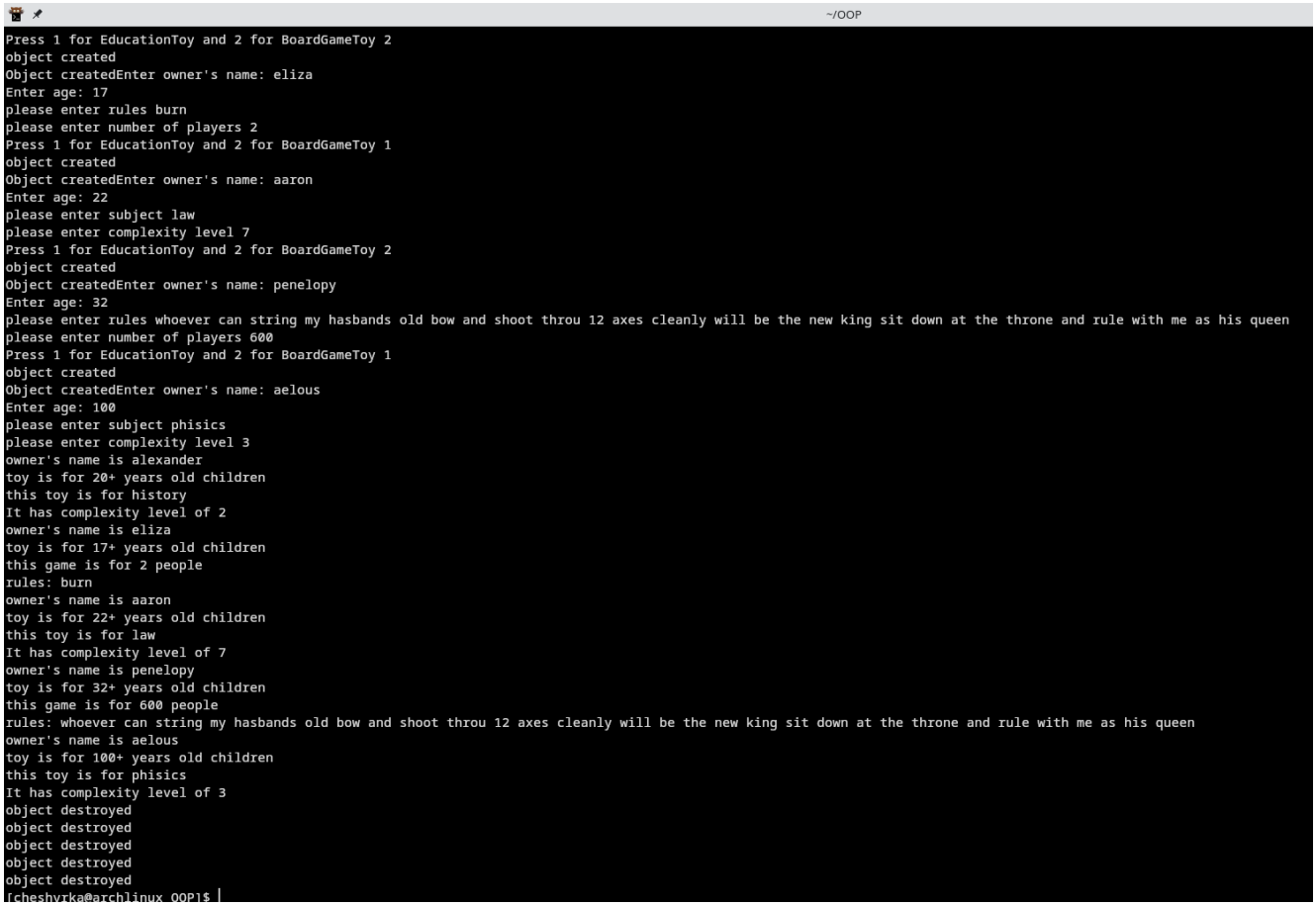
```
    Toy::Toy(const Toy &toy){
        this->ownerName = new char[strlen(toy.ownerName) + 1];
        strcpy(this->ownerName, toy.ownerName);
        this->old = toy.old;
        cout << "object copied " << endl;
    }


    void Toy::setOwnerName(char * ownerName){
        delete[] this->ownerName;
        this->ownerName = new char[strlen(ownerName) + 1];
        strcpy(this->ownerName, ownerName);
    }

    void Toy::setOld(int old){
        this->old = old;
    }

    void Toy::input(){
        char buffer[100];
        cout << "Enter owner's name: ";
        cin.ignore();
        cin.getline(buffer, 100);

        delete[] this->ownerName;
        this->ownerName = new char[strlen(buffer) + 1];
        strcpy(this->ownerName, buffer);

        cout << "Enter age: ";
        cin >> this->old;
    }

    char * Toy::getOwnerName(){
        return ownerName;
    }

    int Toy::getOld(){
        return old;
    }

    void Toy::print() const{
        cout << "owner's name is " << ownerName << endl;
        cout << "toy is for " << old << "+ years old children " << endl;
    }

    Toy::~Toy(){
        delete[] ownerName;
        cout << "object destroyed" << endl;
    }
// EducationToy
    class EducationToy:public Toy {
        string subject;
        int complexityLevel;
    public:
        EducationToy(); //+
```

```
    EducationToy(string subject, int complexityLevel); //+
    EducationToy(const EducationToy &toy);//+
    void input();//+
    void print() const;//+
    void setSubject(string subject);//+
    void setComplexityLevel(int complexityLevel);//+
    string getSubject();//+
    int getComplexityLevel();//+
    ~EducationToy();

};

EducationToy::EducationToy(){
    subject = "";
    complexityLevel = 0;
    cout << "Object created";
}

EducationToy::EducationToy(string subject, int complexityLevel){
    this->subject = subject;
    this->complexityLevel = complexityLevel;
}

EducationToy::EducationToy(const EducationToy &toy){
    this->subject = toy.subject;
    this->complexityLevel = toy.complexityLevel;
}

void EducationToy::input(){
    Toy::input();
    cout<<"please enter subject ";
    cin.ignore();
    getline(cin, subject);
    cout<<"please enter complexity level ";
    cin >> complexityLevel;
}

void EducationToy::print() const{
    Toy::print();
    cout << "this toy is for " << subject << endl;
    cout << "It has complexity level of " << complexityLevel << endl;
}

void EducationToy::setSubject(string subject){
    this->subject = subject;
}

void EducationToy::setComplexityLevel(int complexityLevel){
    this->complexityLevel = complexityLevel;
}

int EducationToy::getComplexityLevel(){
    return complexityLevel;
}

string EducationToy::getSubject(){
    return subject;
```

```
    }

    EducationToy::~EducationToy(){
        cout << "Object destroyed";
    }
//BoardGameToy
    class BoardGameToy:public Toy {
        int numberOfPlayers;
        string rules;
    public:
        BoardGameToy();
        BoardGameToy(int numberOfPlayers, string rules);
        BoardGameToy(const BoardGameToy&);
        void input();
        void print() const;
        void setNumberOfPlayers(int numberOfPlayers);
        void setRules(string rules);
        int getNumberOfPlayers();
        string getRules();
        ~BoardGameToy();

    };


    BoardGameToy::BoardGameToy(){
        rules = " ";
        numberOfPlayers = 0;
        cout << "Object created";
    }

    BoardGameToy::BoardGameToy(int numberOfPlayers, string rules){
        this->rules = rules;
        this->numberOfPlayers = numberOfPlayers;
    }

    BoardGameToy::BoardGameToy(const BoardGameToy &toy){
        this->rules = toy.rules;
        this->numberOfPlayers = toy.numberOfPlayers;
    }

    void BoardGameToy::input(){
        Toy::input();
        cout<<"please enter rules ";
        cin.ignore();
        getline(cin, rules, '\n');
        cout<<"please enter number of players ";
        cin >> numberOfPlayers;
    }

    void BoardGameToy::print() const{
        Toy::print();
        cout << "this game is for " << numberOfPlayers << " people"<<endl;
        cout << "rules: " << rules << endl;
    }

    void BoardGameToy::setRules(string rules){
        this->rules = rules;
```

```
    }

    void BoardGameToy::setNumberOfPlayers(int numberOfPlayers){
        this->numberOfPlayers = numberOfPlayers;
    }

    int BoardGameToy::getNumberOfPlayers(){
        return numberOfPlayers;
    }

    string BoardGameToy::getRules(){
        return rules;
    }

    BoardGameToy::~BoardGameToy(){
        cout << "Object destroyed";
    }

//main
    int main()
    {
        //ввід
        Toy *toys[n];
        for (int i = 0; i < n; ++i)
        {
            int choice;
            Toy* pointerToToy;
            cout<<"Press 1 for EducationToy and 2 for BoardGameToy ";
            cin>>choice;
            //вибрати аудіо чи паперова книига
            switch(choice){
                case 1:
                    {
                        EducationToy* pointerToEducationToy = new EducationToy();
                        pointerToEducationToy->input();
                        pointerToToy = pointerToEducationToy;
                        break;
                    }
                case 2:
                    {
                        BoardGameToy* pointerToBoardGameToy = new BoardGameToy();
                        pointerToBoardGameToy->input();
                        pointerToToy = pointerToBoardGameToy;
                        break;
                    }
                default:
                    {
                        cout << "Invalid choice. Creating EducationToy by default.
\n";
                        pointerToToy = new EducationToy();
                        pointerToToy->input();
                        break;
                    }
            }


            toys[i] = pointerToToy;
```

```
    }
//вивід
    for (int i = 0; i < n; ++i)
    {
        toys[i]->print();
    }



    //звільнення динамічної пам'яті
    for (int i = 0; i < n; ++i)
    {
        delete toys[i];
    }
    return 0;
}

  //done
```



Висновок: в ході виконання лабораторної роботи я Практично ознайомитись з поняттям поліморфізму, його застосуванням та вивчити механізм його реалізації за допомогою віртуальних методів