

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()`.

3. Напишіть програму згідно завдання 2.

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:

```
#ifndef PUBLICATION_H
#define PUBLICATION_H
#include <string>

class Publication
{
```

```

private:
    std::string m_name;
int m_price;
public:
    Publication();
    Publication(std::string a, int pr);

    std::string getName();    int getPrice();
    void setName();
        void setPrice();

    void setName(std::string);
        void setPrice(int);    void setAll();
    void showPublication(); };

```

```

class Book : public Publication
{
private:
    int m_NumOfPages;

public:
    Book();
    Book(std::string name, int price, int num );

int getNumOfPages();
    void setNumOfPages();
void setNumOfPages(int);
void setBook();
    void showBook(); };

```

```

class Type : public Publication
{

```

```

private:
    struct Time{
        int m_minute;
        int m_hour;
        int m_day;
        int m_mounth;
        int m_year;    };
    Time m_time;
public:
    Type();
    Type(std::string name, int price);
    void setTime();
    void setType();
    void getTime();
    void showType(); };
#endif // PUBLICATION_H

```

```

#include<iostream>
#include <string>
#include <publication.h>

Publication::Publication() : m_name("Unknown"),m_price(0){}

Publication::Publication(std::string a, int pr)
    : m_name(a),m_price(pr){ }
std::string Publication::getName() { return m_name;}
int Publication::getPrice() { return m_price; }

```

```

void Publication::setName(std::string name)
{ m_name = name;}

void Publication::setPrice(int price){ m_price = price;}

void Publication::setName() {
    std::cout << "Enter a name of book: ";
    std::cin >> m_name; }

void Publication::setPrice() {
    std::cout << "Enter price";
    std::cin >> m_price; }

void Publication::setAll(){    setName();    setPrice(); }

void Publication::showPublication()
{
    std::cout << "Name: " << getName() << std::endl;
    std::cout << "Price: " << getPrice() << std::endl; }

```

```

Book::Book() :Publication(), m_NumOfPages(0) {}

Book::Book(std::string name, int price, int num) :
Publication(name, price), m_NumOfPages(num) {}

int Book::getNumOfPages() { return m_NumOfPages; }

void Book::setNumOfPages() {
    std::cout << "Enter number of pages: ";
    std::cin >> m_NumOfPages; }

void Book::setNumOfPages(int n){ m_NumOfPages = n; }

void Book::setBook(){    setAll();    setNumOfPages(); }

void Book::showBook(){    showPublication();

    std::cout << "Number of pages: " << getNumOfPages() <<
    std::endl; }

```

```

Type::Type():Publication() {}

Type::Type(std::string name, int price)
    : Publication(name,price) { setTime();}

void Type::setTime(){    std::cout << "Enter a minute";

std::cin >> m_time.m_minute;    std::cout << "hour:";
std::cin >> m_time.m_hour;                                std::cout
<< "day:";    std::cin >> m_time.m_day;

std::cout << "mounth:";    std::cin >> m_time.m_mounth;
std::cout << "year";

std::cin >> m_time.m_year; }

void Type::setType(){    setAll();    setTime(); }

void Type::getTime( ){

    std::cout << "Time: " << m_time.m_minute << "." <<
m_time.m_hour << "." << m_time.m_day << "." <<
m_time.m_mounth << "." << m_time.m_year << std::endl; }

void Type::showType(){    showPublication();
    getTime(); }

int main() {

    Book book;

    book.setNumOfPages(10);

    book.showBook();
    Type type("Ann",15);

    type.showType();

}

```

Результат виконання програми:

```

Name: Unknown
Price: 0
Number of pages: 10
Enter a minute10
hour:9
day:12
mounth:11
year2020
Name: Ann
Price: 15
Time: 10.9.12.11.2020

```

Завдання 3: 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:

```

#ifndef PUBLICATION_H #define PUBLICATION_H
#include <string>
class Publication
{
private:
    std::string m_name;
    int m_price;
public:
    Publication();
    Publication(std::string a, int pr);

    std::string getName();

```

```

    int getPrice();
void setName();

    void setPrice();
void setName(std::string);

    void setPrice(int);

    void setAll();
void showPublication(); };

class Sales{ private:    float m_priceInMonth[3] = {};
public:    Sales();

        Sales(float a, float b, float c);
void setPriceM();

        void setPriceM(float a,float b,float c);
        void getPriceM();    void showSales(); };

class Book : public Publication, public Sales
{ private:    int m_NumOfPages;
public:    Book();

        Book(std::string name, int price, int num );

        Book(std::string name, int price, int num, float a,
float b, float c );

        int getNumOfPages();

void setNumOfPages();

        void setNumOfPages(int);

        void setBook();
void showBook(); };

class Type : public Publication, public Sales
{
private:

```

```

    struct Time
{
    int m_minute;
    int m_hour;
    int m_day;
    int m_mounth;
    int m_year;    };
    Time m_time;
public:
    Type();
    Type(std::string name, int price);
    Type(std::string name, int price, float a, float b,
float c );
    void setTime();    void setType();    void getTime();
    void showType(); };
#endif // PUBLICATION_H

#include<iostream> #include <string> #include <publication.h>

Publication::Publication() : m_name("Unknown"),m_price(0){}

Publication::Publication(std::string a, int pr)
: m_name(a),m_price(pr){ }

std::string Publication::getName() { return m_name;}
int Publication::getPrice() { return m_price; }

void Publication::setName(std::string name)
{ m_name = name;}

void Publication::setPrice(int price){ m_price = price;}

```



```

void Publication::setName()
{
    std::cout << "Enter a name of book: ";
    std::cin >> m_name;
}

void Publication::setPrice()
{
    std::cout << "Enter price";    std::cin >> m_price;
}

void Publication::setAll(){
    setName();
    setPrice(); }

void Publication::showPublication()
{
    std::cout << "Name: " << getName() << std::endl;
    std::cout << "Price: " << getPrice() << std::endl;
}

```

```

Sales::Sales(){} Sales::Sales(float a, float b, float c)
{setPriceM( a,b,c);}

```

```

void Sales::setPriceM(){
    for (int i = 0;i < 3;i++)
    {
        std::cout << "Enter price 1:" ;
        std::cin >> m_priceInMonth[i];
    }
}

```

```

void Sales::setPriceM(float a, float b, float c){
m_priceInMonth[0] = a;
    m_priceInMonth[1] = b;
}

```

```

        m_priceInMonth[2] = c; }

void Sales::getPriceM(){    for (int i = 0;i<3;i++)
{
    std::cout << "Price " << i+1 << ": " <<
m_priceInMonth[i] << std::endl;    }} void
Sales::showSales(){ getPriceM(); }

Book::Book() :Publication(),Sales(), m_NumOfPages(0) {}
Book::Book(std::string name, int price, int num) :
Publication(name, price), m_NumOfPages(num) {}
Book::Book(std::string name, int price, int num, float a,
float b, float c) : Publication(name, price),Sales(a,b,c),
m_NumOfPages(num) {} int Book::getNumOfPages() { return
m_NumOfPages; }
void Book::setNumOfPages() {    std::cout << "Enter number
of pages: ";    std::cin >> m_NumOfPages; } void
Book::setNumOfPages(int n){ m_NumOfPages = n; } void
Book::setBook(){    setAll();    setNumOfPages(); } void
Book::showBook(){    showPublication();    showSales();
std::cout << "Number of pages: " << getNumOfPages() <<
std::endl; }

Type::Type():Publication(), Sales(){}
Type::Type(std::string name, int price) :
Publication(name,price) { setTime();}

Type::Type(std::string name, int price, float a, float b,
float c ):Publication(name,price),Sales(a,b,c)
{ setTime();}

void Type::setTime(){
    std::cout << "Enter a minute";    std::cin >>
m_time.m_minute;

    std::cout << "hour:";    std::cin >> m_time.m_hour;

    std::cout << "day:";

```

```

    std::cin >> m_time.m_day;

    std::cout << "mounth:";

    std::cin >> m_time.m_mounth;
std::cout << "year";

    std::cin >> m_time.m_year; }

void Type::setType(){    setAll();    setTime(); }
void Type::getTime( )

{    std::cout << "Time: " << m_time.m_minute << "." <<
m_time.m_hour << "." << m_time.m_day << "." <<
m_time.m_mounth << "." << m_time.m_year << std::endl; }

void Type::showType(){    showPublication();    showSales();
getTime(); }

int main() {
Book book("Puppies", 10, 56, 4.0, 1.0, 2.0);

    book.setNumOfPages(10);

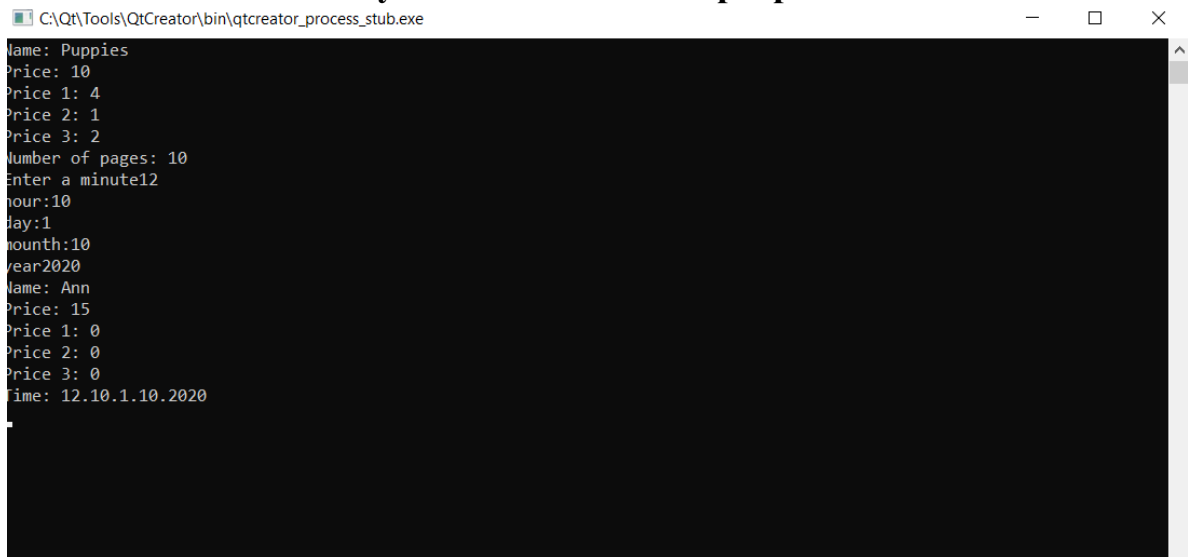
    book.showBook();

    Type type("Ann",15);

    type.showType();
}

```

Результат виконання програми:

A screenshot of a Qt Creator console window. The title bar shows the file path 'C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe'. The console output is as follows:

```
Name: Puppies
Price: 10
Price 1: 4
Price 2: 1
Price 3: 2
Number of pages: 10
Enter a minute12
hour:10
day:1
month:10
year2020
Name: Ann
Price: 15
Price 1: 0
Price 2: 0
Price 3: 0
Time: 12.10.1.10.2020
```

Завдання 4:

```
#ifndef TWONUMBERS_H
#define TWONUMBERS_H
#include <iostream>
class TwoNumbers
{
private:
    int m_NumOne;
    int m_NumTwo;
public:
    TwoNumbers();
    TwoNumbers(int a,int b);
    void setNumOne(int a);
    void setNumTwo(int a);
    void setTwoNumbers(int a,int b);
    int getNumOne();
    int getNumTwo();
    void showTwoNumbers();
};
```

```

    ~TwoNumbers(); };
#endif // TWONUMBERS_H

#include "twonumbers.h"
TwoNumbers::TwoNumbers(): m_NumOne(0), m_NumTwo(0){}

TwoNumbers::TwoNumbers(int a, int b): m_NumOne(a),
m_NumTwo(b){}

void TwoNumbers::setNumOne(int a){ m_NumOne = a;}
void TwoNumbers::setNumTwo(int a){ m_NumTwo = a;}
void TwoNumbers::setTwoNumbers(int a,int b)
{m_NumOne = a; m_NumTwo = b;}
int TwoNumbers::getNumOne(){ return m_NumOne; }
int TwoNumbers::getNumTwo(){ return m_NumTwo; }
void TwoNumbers::showTwoNumbers()
{
std::cout << "First: " << m_NumOne << std::endl;
std::cout << "Second: " << m_NumTwo << std::endl; }
TwoNumbers::~~TwoNumbers(){}

#ifndef GEOMETRYPROGRESSION_H
#define GEOMETRYPROGRESSION_H

#include <twonumbers.h>
class GeometryProgression : public TwoNumbers
{
private:    int m_firstNumber;
           int m_indexOfProgression;
public:    GeometryProgression();
           GeometryProgression(int num, int index);
           GeometryProgression(int NumOne, int NumTwo, int num, int
index);

```

```

    int sumOfProgression(int n);
    int getElementOfProgression(int n); //елемент прогресії з
індексом n    int getFirstNumber() const;

    int getIndex() const;

    void showProgression(int n);

    void setFirstNumber(int a);

    void setIndex(int a);
~GeometryProgression(); };
#endif // GEOMETRYPROGRESSION_H

```

```

#include "geometryprogression.h"

```

```

#include <math.h>

```

```

GeometryProgression::GeometryProgression()

```

```

: TwoNumbers(), m_firstNumber(1),m_indexOfProgression(1)
{}

```

```

GeometryProgression::GeometryProgression(int num, int
index): TwoNumbers(), m_firstNumber(num),
m_indexOfProgression(index){}

```

```

GeometryProgression::GeometryProgression(int NumOne, int
NumTwo, int num, int index) : TwoNumbers(NumOne,
NumTwo), m_firstNumber(num), m_indexOfProgression(index){}

```

```

int GeometryProgression::sumOfProgression(int n){    int sum
= 0;    for (int i = 1; i <= n; i++){

        sum += getElementOfProgression(i);    }

    return sum; }

```

```

int GeometryProgression::getElementOfProgression(int n)

```

```

{    int element = m_firstNumber *
pow(m_indexOfProgression, n-1);    return element; }

```

```

int GeometryProgression::getFirstNumber() const

```

```

{ return m_firstNumber; }

```

```

int GeometryProgression::getIndex() const
{ return m_indexOfProgression; }

void GeometryProgression::showProgression(int n) {
    for (int i = 1; i <= n; i++)
    {
        std::cout << "Element " << i << ": " <<
        getElementOfProgression(i) << std::endl;    } }
void GeometryProgression::setFirstNumber(int a)
{ m_firstNumber = a; } void
GeometryProgression::setIndex(int a)
{ m_indexOfProgression = a; }

GeometryProgression::~GeometryProgression(){}

#include <QCoreApplication>
#include <geometryprogression.h>
#include <iostream>

int main() {
    GeometryProgression a, b(10,9,1,2);
    a.setFirstNumber(2);
    a.setIndex(4);
    std::cout << "Progesion a: " << std::endl;
    a.showProgression(3);
    std::cout << "Sum of progresion a: "<<
a.sumOfProgression(3);
    std::cout << "Third element of b: " << b.getIndex() <<
std::endl;    std::cout << "First element of b: " <<
b.getFirstNumber() << std::endl;
    b.showTwoNumbers(); }

```

Результат виконання програми:

```
C:\Users\user\Documents\generator\generator_process.py
Progression a:
Element 1: 2
Element 2: 8
Element 3: 32
Sum of progression a: 42Third element of b: 2
First element of b: 1
First: 10
Second: 9
```

Висновок: Я вивчила успадкування класів.