Лабораторна робота №4

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.

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*.

Код програми:

“Publication.h”:

```
#ifndef PUBLICATION_H
#define PUBLICATION_H
```

```
#include <iostream>
class Publication
{
private:
    std::string m_name;
   float m_price;
public:    Publication();
   Publication(std::string a, int pr);
 std::string getName();
    float getPrice();
    virtual bool itsOverSize(){ return false;}
    virtual void setInfo(){        std::cout << "Error";
}
 void setName();
    void setPrice();
    void setName(std::string);
    void setPrice(int);
    void setParent();
    virtual void showObject(){
        std::cout << "Function not found!!!!\n";    }
    void showPublication();    virtual ~Publication(){} };
#endif // PUBLICATION_H
```

“Publication.cpp”:

```
#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;}
```

```
float  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 << "Name: ";
    std::cin >> m_name; }
void Publication::setPrice()
{    std::cout << "Enter price";
    std::cin >> m_price; }
void Publication::setParent()
{
   setName();
    setPrice(); }
void Publication::showPublication(){
std::cout << "Name: " << getName() << std::endl;
std::cout << "Price: " << getPrice() << std::endl; }
```

“Book.h”:

```
#ifndef BOOK_H
#define BOOK_H
#include<publication.h>
class Book :
public Publication
{
private:
    int m_NumOfPages;
```

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

  int getNumOfPages();
virtual bool itsOverSize(){
        if(getNumOfPages()> 800)
{        std::cout << "It'ss oversize!!!!!!!!!!!!!!!";
        return true;}
     return false; }
   void setNumOfPages();
    void setNumOfPages(int);
  virtual void setInfo()
 {        setParent();
        setNumOfPages();    }
    virtual void showObject(){
        showPublication();
        std::cout << "Number of pages: " << getNumOfPages()
<< std::endl;    }
    virtual ~Book(){} };

#endif // BOOK_H
```

“Book.cpp”:

```
#include "book.h"
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; }
```

“Tape.h”:

```
#ifndef TAPE_H
#define TAPE_H
#include <publication.h>
class Tape : public Publication
{ private:    int m_index;
public:    Tape();
   Tape(std::string name, int price, int num );
 bool itsOverSize()
{        if(getIndex()> 90){
            std::cout << "It'ss oversize!!!!!!!!!!!!!!!";
            return true;        }
        return false;}
    int getIndex();
 void setIndex();
    void setIndex(int);
   virtual void  setInfo(){      setParent();
setIndex();    }
    virtual void showObject()
{        showPublication();
        std::cout << "Index: " << getIndex() << std::endl;
}    virtual ~Tape(){} };
#endif // TAPE_H
```

“tape.cpp”:

```
#include "tape.h"
 Tape::Tape():Publication(),m_index(0){}
Tape::Tape(std::string name, int price, int num)
```

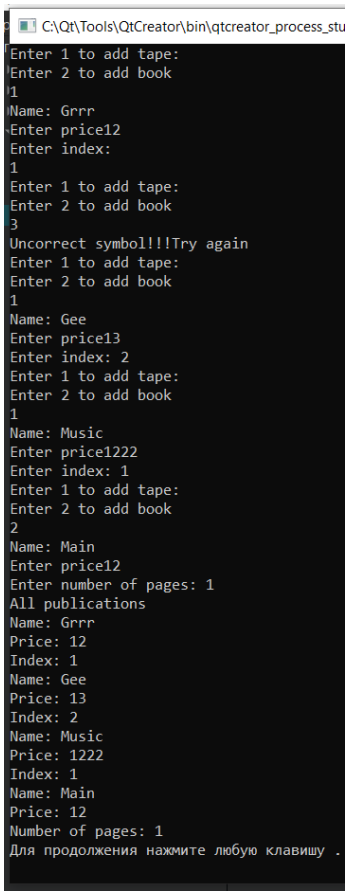
```
 : Publication(name, price), m_index(num)  {}
int Tape::getIndex()  { return m_index; }
void Tape::setIndex()
{    std::cout << "Enter index: ";
    std::cin >> m_index; }
void Tape::setIndex(int n){ m_index = n; }
```

“Main.cpp:”

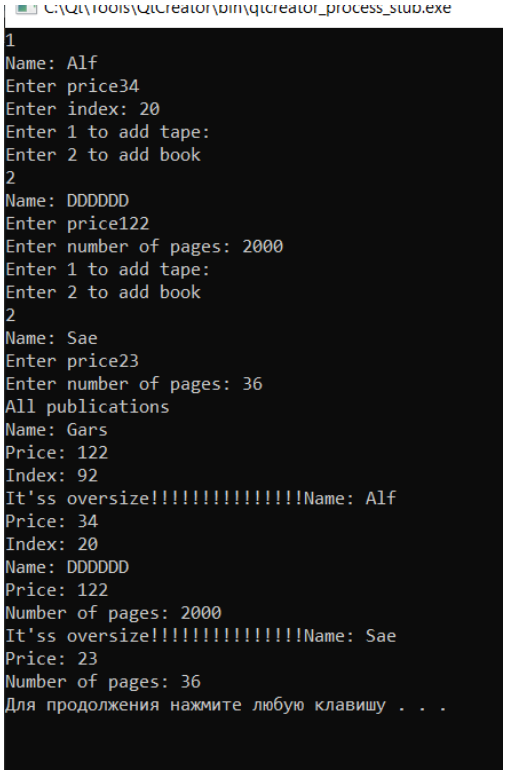
```
#include <tape.h> #include <book.h>
int main() { int s=0; int lenght = 4;
Publication *p[4]; bool a = true; char n; while(a){
std::cout << "Enter 1 to add tape:\n";   std::cout <<
"Enter 2 to add book\n";
 std::cin >> n;   switch (n) {   case '1':
   p[s] = new Tape;       p[s]->setInfo();       s++;
break;   case '2':       p[s] = new Book;       p[s]-
>setInfo();       s++;       break;   default:
std::cout << "Uncorrect symbol!!!Try again\n";
 }  if(s== lenght)      break;
} std::cout << "All publications\n"; for (int i
=0;i<lenght;i++){   p[i]->showObject();   p[i]-
>itsOverSize(); } // знищити всі фігури for(int i=0;
i<lenght; i++) delete p[i];
system("pause");
return 0; }
```

Результат виконання роботи:

Завдання 1:


C:\Qt\Tools\QtCreator\bin\qtcreator_process_stu
Enter 1 to add tape:
Enter 2 to add book
1
Name: Grrr
Enter price12
Enter index:
1
Enter 1 to add tape:
Enter 2 to add book
3
Uncorrect symbol!!!Try again
Enter 1 to add tape:
Enter 2 to add book
1
Name: Gee
Enter price13
Enter index: 2
Enter 1 to add tape:
Enter 2 to add book
1
Name: Music
Enter price1222
Enter index: 1
Enter 1 to add tape:
Enter 2 to add book
2
Name: Main
Enter price12
Enter number of pages: 1
All publications
Name: Grrr
Price: 12
Index: 1
Name: Gee
Price: 13
Index: 2
Name: Music
Price: 1222
Index: 1
Name: Main
Price: 12
Number of pages: 1
Для продолжения нажмите любую клавишу .

Завдання 2:


C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe
1
Name: Alf
Enter price34
Enter index: 20
Enter 1 to add tape:
Enter 2 to add book
2
Name: DDDDDD
Enter price122
Enter number of pages: 2000
Enter 1 to add tape:
Enter 2 to add book
2
Name: Sae
Enter price23
Enter number of pages: 36
All publications
Name: Gars
Price: 122
Index: 92
It'ss oversize!!!!!!!!!!!!!!Name: Alf
Price: 34
Index: 20
Name: DDDDDD
Price: 122
Number of pages: 2000
It'ss oversize!!!!!!!!!!!!!!Name: Sae
Price: 23
Number of pages: 36
Для продолжения нажмите любую клавишу . . .

Завдання 3: Створити клас ПАРА ЧИСЕЛ. Визначити віртуальну функцію обчислення суми цих чисел. Створити похідні класи ГЕОМЕТРИЧНА ПРОГРЕСІЯ, АРИФМЕТИЧНА ПРОГРЕСІЯ з полями: перший та п’ятий елементи прогресії та своїми функціями обчислення суми п’яти елементів прогресії. Для перевірки використати масив вказівників на об’єкти базового класу, яким присвоїти адреси об’єктів похідних класів.

Код програми:

```
#include<iostream>
#include<math.h>
class TwoNumbers
{
private:
   int m_NumOne;
   int m_NumTwo;
public:
   TwoNumbers ();
   TwoNumbers (int a, int b);
   virtual void setInfo () {}
   void setNumOne (int a);
   void setNumTwo (int a);
   void setTwoNumbers (int a, int b);
   virtual int sumOfProgression (int n) { return n; }
   int getNumOne ();
   int getNumTwo ();
   void showTwoNumbers ();
   virtual ~TwoNumbers ();
```

```
};


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 () {}

class GeometryProgression : public TwoNumbers
{
private:
   int m_firstNumber;
   int m_fifthnum;
public:
```

```
    GeometryProgression ();
    GeometryProgression (int num, int);
    GeometryProgression (int NumOne, int NumTwo, int num, int);
    virtual int sumOfProgression (int n)
    {
        int sum = 0;
        for( int i = 1; i <= n; i++ ) {
            sum += getElementOfProgression (i);
        }
        return sum;
    }
    virtual void setInfo ()
    {
        int k;
        std::cout << "Enter first num:";
        std::cin >> k;
        setFirstNumber (k);
        std::cout << "Enter fifth num:";
        std::cin >> k;
        setFifthNum (k);
    }
    double getindex ();
    double getElementOfProgression (int n); //елемент прогресії з індексом
n
    int getFirstNumber () const;
    int getFifthNum () const;
    void showProgression (int n);
```

```
    void setFirstNumber (int a);
    void setFifthNum (int a);

    virtual ~GeometryProgression ();
};

GeometryProgression::GeometryProgression () : TwoNumbers (),
m_firstNumber (1), m_fifthnum (1) {}
GeometryProgression::GeometryProgression (int num, int i) : TwoNumbers
(), m_firstNumber (num), m_fifthnum (i) {}
GeometryProgression::GeometryProgression (int NumOne, int NumTwo, int
num, int i)
    : TwoNumbers (NumOne, NumTwo), m_firstNumber (num), m_fifthnum (i)
{
}


double GeometryProgression::getindex ()
{
    return pow (m_fifthnum / m_firstNumber, 1 / 4);
}
double GeometryProgression::getElementOfProgression (int n)
{
    int element = m_firstNumber * pow (getindex (), n - 1);
    return element;
}
int GeometryProgression::getFirstNumber () const { return m_firstNumber; }
int GeometryProgression::getFifthNum () const { return m_fifthnum; }
```

```
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::setFifthNum (int a) { m_fifthnum = a; }

GeometryProgression::~GeometryProgression () {}


class AlgebraProgression :public TwoNumbers
{
private:
   int m_firstNumber;
   int m_fifthnum;
public:
   AlgebraProgression ();
   AlgebraProgression (int num, int);
   AlgebraProgression (int NumOne, int NumTwo, int num, int);
   virtual int sumOfProgression (int n)
   {
      int s = 0;
      for( int i = 1; i <= n; i++ ) {
```

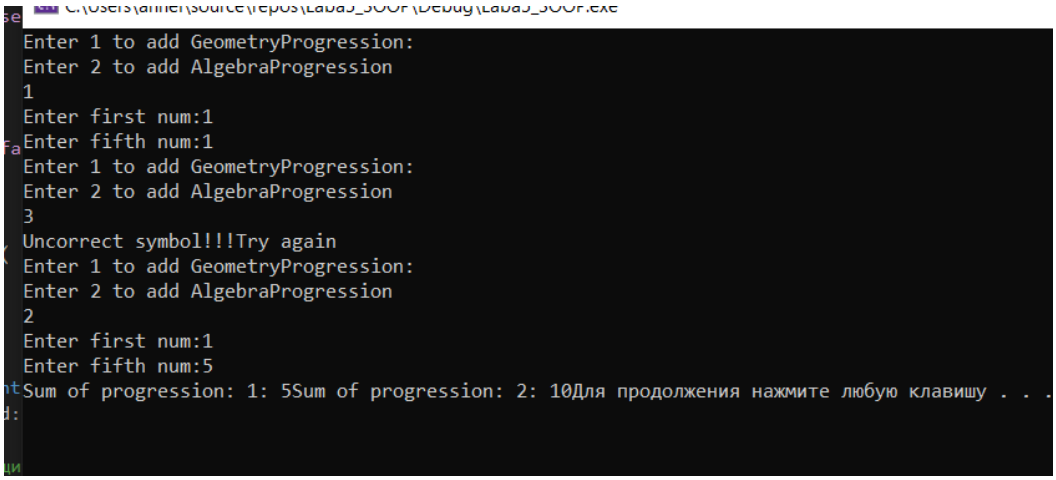
```
        s += getElementOfProgression (i);
      }
      return s;
   }
   virtual void setInfo ()
   {
      int k;
      std::cout << "Enter first num:";
      std::cin >> k;
      setFirstNumber (k);
      std::cout << "Enter fifth num:";
      std::cin >> k;
      setFifthNum (k);
   }
   int getindex ();
   int getElementOfProgression (int n); //елемент прогресії з індексом n
   int getFirstNumber () const;
   int getFifthNum () const;
   void showProgression (int n);
   void setFirstNumber (int a);
   void setFifthNum (int a);
};


AlgebraProgression::AlgebraProgression () :TwoNumbers (), m_firstNumber
(0), m_fifthnum (1) {}
```

```
AlgebraProgression::AlgebraProgression (int num, int i) : TwoNumbers (), m_firstNumber (num), m_fifthnum (i) {}

AlgebraProgression::AlgebraProgression (int NumOne, int NumTwo, int num, int i) : TwoNumbers (NumOne, NumTwo), m_firstNumber (num), m_fifthnum (i) {}


int AlgebraProgression::getindex ()

{

   return (m_fifthnum - m_firstNumber) / 4;

}

int AlgebraProgression::getElementOfProgression (int n) //елемент прогресії з індексом n

{

   return m_firstNumber + n * getindex ();

}

int AlgebraProgression::getFirstNumber () const { return m_firstNumber; }

int AlgebraProgression::getFifthNum () const { return m_fifthnum; }

void AlgebraProgression::showProgression (int n)

{

   for( int i = 1; i <= n; i++ ) {

      std::cout << "Element " << i << ": " << getElementOfProgression (i) << std::endl;

   }

}

void AlgebraProgression::setFirstNumber (int a) { m_firstNumber = a; }

void AlgebraProgression::setFifthNum (int a) { m_firstNumber = a; }
```

```
int main ()
{
   int s = 0;
   int lenght = 2;

   TwoNumbers *p[2];
   bool a = true;
   char n;
   while( a ) {
      std::cout << "Enter 1 to add GeometryProgression:\n";
      std::cout << "Enter 2 to add AlgebraProgression\n";

      std::cin >> n;
      switch( n ) {
      case '1':

         p[s] = new GeometryProgression;
         p[s]->setInfo ();
         s++;
         break;
      case '2':
         p[s] = new AlgebraProgression;
         p[s]->setInfo ();
         s++;
         break;
      default:
         std::cout << "Uncorrect symbol!!!Try again\n";
```

```
        }
        if( s == lenght )
            break;

    }

    for( int i = 0; i < lenght; i++ ) {
        std::cout << "Sum of progression: " << i + 1 << ": " << p[i]-
>sumOfProgression (5);
    }

    for( int i = 0; i < lenght; i++ )
        delete p[i];

    system ("pause");

}
```

Результат виконання програми:



Висновок: Я вивчила віртуальні функції та поліморфізм.