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

Book.h

```
#ifndef BOOK_H
#define BOOK_H
#include "publication.h"
using namespace std;

class book : public publication
{
private:
   int page;
public:
   book();
```

```
   void getdata() override;
   void putdata() override;
};

#endif // BOOK_H
```

.....

```
#ifndef PUBLICATION_H
#define PUBLICATION_H
#include <string>
using namespace std;

class publication
{
private:
   string name;
   float cost;

public:
   publication();
   virtual void getdata();
   virtual void putdata();
   //virtual  ~publication();
};

#endif // PUBLICATION_H
```

Publication.h

.....

```
#ifndef TYPE_H
#define TYPE_H
#include "publication.h"
using namespace std;

class type : public publication
{
private:

   float min;
public:
   type();
   void getdata() override;
   void putdata() override;
};

#endif // TYPE_H
```

Type.h

.....

```
#include "book.h"
#include <iostream>
using namespace std;
book::book()
{

}
```

Book.cpp

```
void book::getdata()
{
   publication::getdata();
   cout<<"Enter Page=";
   cin>>page;
}

void book::putdata()
{
   publication::putdata();
   cout<<"Page="<<page<<endl;
}
```

.....

```
#include "publication.h"
#include <iostream>
using namespace std;
publication::publication()
{

}

void publication::getdata()
{
   cout<<"Enter Name\n";
   cin>>name;
   cout<<"Enter Cost\n";
   cin>>cost;
}

void publication::putdata()
{
   cout<<"Name="<<name<<"\n"<<"Cost="<<cost<<endl;

}
```

Publication.cpp

.....

```
#include "type.h"
#include <iostream>
using namespace std;
type::type()
{

}

void type::getdata()
{
    publication::getdata();
    cout<<"Enter Min=";
    cin>>min;

}

void type::putdata()
{
    publication::putdata();
    cout<<"time="<<min<<endl;
}
```

Type.cpp

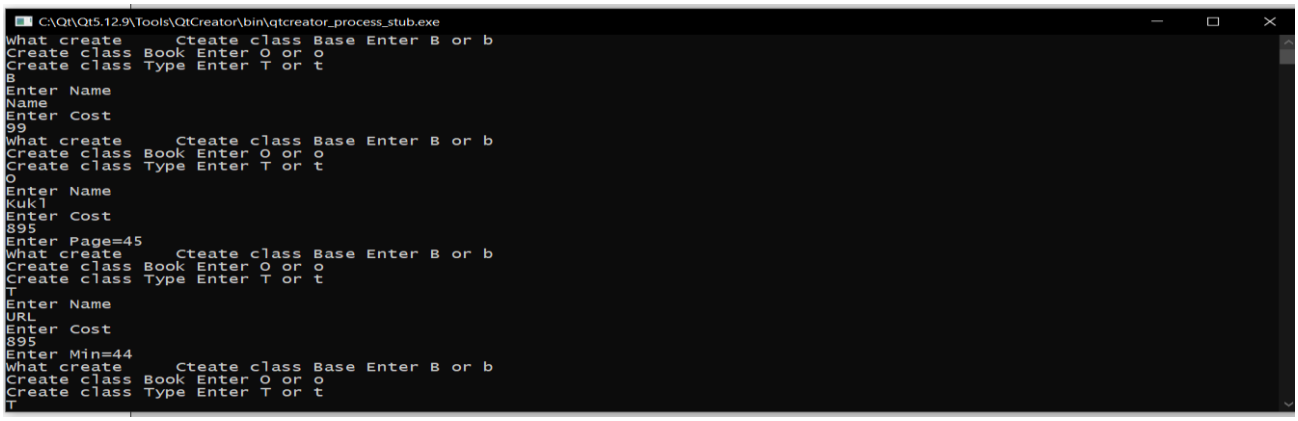
.....

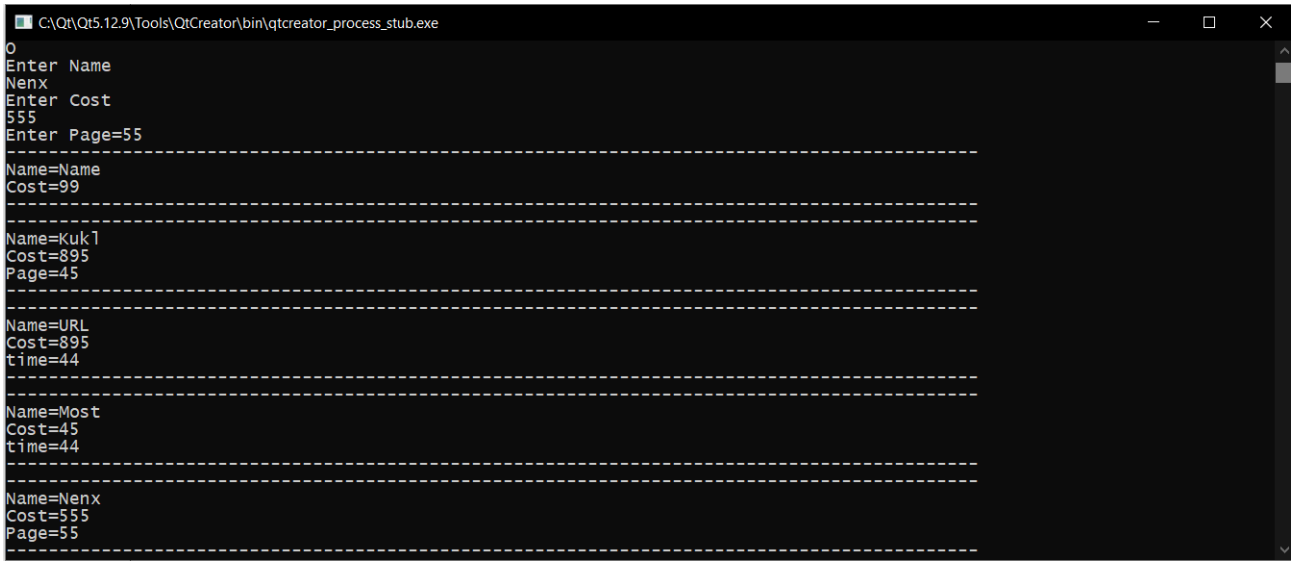
```
#include <iostream>
#include "publication.h"
#include "book.h"
#include "type.h"
```

main.cpp

```
using namespace std;

int main()
{   int n=4;
    publication *arr[n];
    int i=0;
    while(i<=n)
    {
        char a;
        cout<<"What create\t";
        cout<<"Cteate class Base Enter B or b\nCreate class Book Enter O or o\nCreate class Type Enter T or
t\n";
        cin>>a;
        if(a=='o'||a=='O') {arr[i]=new book();
            arr[i]->getdata();};
        if(a=='T'||a=='t') {arr[i]=new type();
            arr[i]->getdata();};
        if(a=='B'||a=='b') {arr[i]=new publication();
            arr[i]->getdata();};
        i++;
    }
    i=0;
    while(i<=n)
    {cout<<"-------------------------------------------------------------------------------------------\n";
        arr[i]->putdata();
        i++;
        cout<<"-------------------------------------------------------------------------------------------\n";
    }

    return 0;
}
```

Результат





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

Book.h

```
#ifndef BOOK_H
#define BOOK_H
#include "publication.h"
using namespace std;

class book : public publication
{
private:
   int page;
public:
   book();
   void getdata() override;
   void putdata() override;
   bool isOversize() override;
   ~book() override;
};

#endif // BOOK_H
```

//

```
#ifndef PUBLICATION_H
#define PUBLICATION_H
#include <string>
using namespace std;

class publication
{
private:
   string name;
   float cost;
public:
   publication();
   virtual void getdata();
   virtual void putdata();
   virtual bool isOversize();
   virtual ~publication();
};

#endif // PUBLICATION_H
```

Publication.h

```
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
#ifndef TYPE_H
#define TYPE_H
#include "publication.h"
using namespace std;

class type : public publication
{
private:
    float min;
public:

    type();
    void getdata() override;
    void putdata() override;
    bool isOversize() override;
    ~type() override;
};

#endif // TYPE_H
```

Type.h

```
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
#include "book.h"
#include <iostream>
using namespace std;
book::book()
{

}

void book::getdata()
{
   publication::getdata();

   cout<<"Enter Page=";
   cin>>page;
}
```

Book.cpp

```
void book::putdata()
{
   publication::putdata();
   cout<<"Page="<<page<<endl;
}

bool book::isOversize()
{
   if(page>800) return true;
   else return false;
}

book::~book()
{

}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
#include "publication.h"
#include <iostream>
using namespace std;
publication::publication()
{

}

void publication::getdata()
{
   cout<<"Enter Name\n";
   cin>>name;
   cout<<"Enter Cost\n";
   cin>>cost;
}

void publication::putdata()
{
   cout<<"Name="<<name<<"\n"<<"Cost="<<cost<<endl;

}

bool publication::isOversize()
{
        return true;

}

publication::~publication()
{

}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
#include "type.h"
#include <iostream>
using namespace std;
```

Publication.cpp

```
type::type()
{

}

void type::getdata()
{
   publication::getdata();
   cout<<"Enter Min=";
   cin>>min;

}

void type::putdata()
{
   publication::putdata();
   cout<<"time="<<min<<endl;
}

bool type::isOversize()
{
   if(min>800) return true;
   else return false;


}

type::~type()
{

}
```

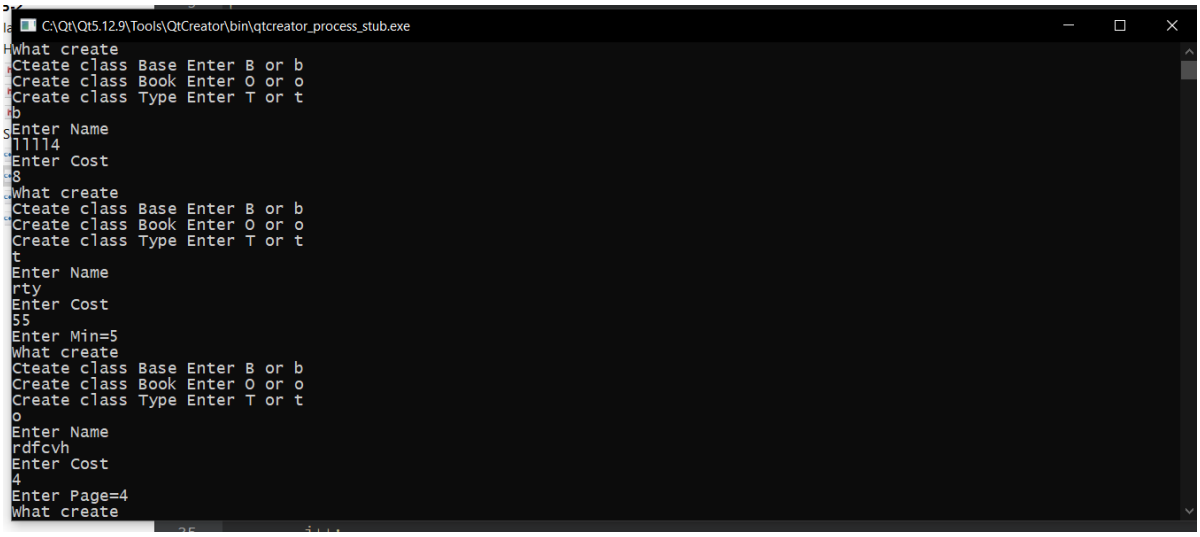
Type.cpp

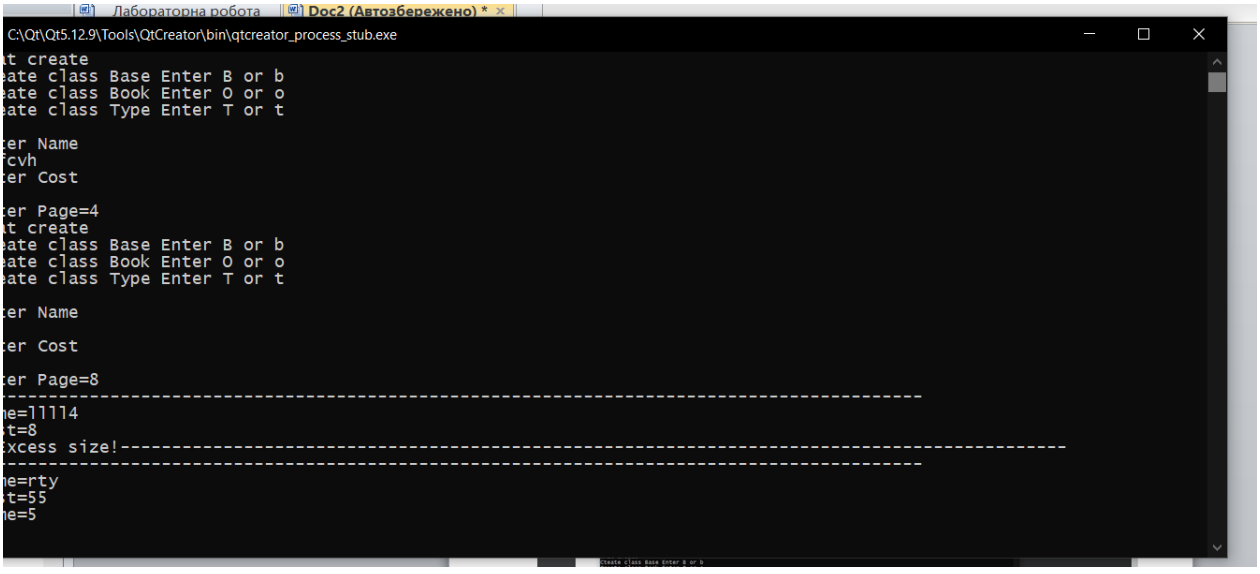
```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
#include <iostream>
#include "publication.h"
#include "book.h"
#include "type.h"

using namespace std;

int main()
{
  //   setlocale(LC_CTYPE, "ukr");
   bool flag=1;
   publication *arr[4];
   int i=0;
   while(i<4)
   {
      char a;
      cout<<"What create\n";
      cout<<"Cteate class Base Enter B or b\nCreate class Book Enter O or o\nCreate class Type Enter T or
t\n";
      cin>>a;
      if(a=='o'||a=='O') {arr[i]=new book();
         arr[i]->getdata();};
```

main.cpp

```
    if(a=='T'||a=='t') {arr[i]=new type();
       arr[i]->getdata();};
    if(a=='B'||a=='b') {arr[i]=new publication();
       arr[i]->getdata();};
    i++;
  }
  i=0;
  while(i<4)
  {cout<<"--------------------------------------------------------------------------------------------\n";
    arr[i]->putdata();
    flag =arr[i]->isOversize();
    if(flag)cout<<"«Excess size!";
    delete *arr;
    i++;
    cout<<"--------------------------------------------------------------------------------------------\n";
  }

  return 0;
}
```

Результат





Завдання 3.

Створити клас ЧЕТВІРКА ЧИСЕЛ. Визначити віртуальну функцію обчислення добутку двох чисел. Створити похідні класи КУБІЧНЕ РІВНЯННЯ, ПРИВЕДЕНЕ КУБІЧНЕ РІВНЯННЯ з полями-коефіцієнтами та своїми функціями обчислення коренів. Для перевірки використати масив вказівників на об’єкти базового класу, яким присвоїти адреси об’єктів похідних класів

Код програми

…… fournubers.h……

```
#ifndef FOURNUBERS_H
#define FOURNUBERS_H
#include <iostream>

using namespace std;

class fournubers
{
   double a,b,c,d;
public:
   fournubers();
   fournubers(double _a,double _b,double _c,double _d):a(_a),b(_b),c(_c),d(_d){};
  virtual void get();
   virtual void math();


};

#endif // FOURNUBERS_H
```

…… kub.h……..

```
#ifndef KUB_H
#define KUB_H
#include "fournubers.h"
#include <math.h>

const double PI = 4.0 * atan( 1.0 );

class kub:public fournubers
{
private:
   double a, b, c, d,p,q;
public:
   kub();
   kub(double _A,double _B,double _C,double _D);
   void math()override;
   virtual void get()override;
```

```
};

#endif // KUB_H
…… cubik_equation h……
#ifndef CUBIK_EQUATION_H
#define CUBIK_EQUATION_H
#include "fournubers.h"
#include <math.h>

const double PII = 4.0 * atan( 1.0 );


class cubik_equation:public fournubers
{
private:
   double a=1, b, c, d,p,q;
public:
    cubik_equation();
   cubik_equation(double b,double c,double d);
   void get() override;
   void math() override;

};

#endif // CUBIK_EQUATION_H
……cubik_equation.cpp……..
#include "cubikequation.h"

cubik_equation::cubik_equation()
{

}

cubik_equation::cubik_equation(double b, double c, double d):fournubers(a=1,b,c,d){}

void cubik_equation::get()
{
cout<<"Введіть коефіцієнти кубічного рфівняння Ax^3+Bx^2+Cx+D"<<endl;
   cout<<"A=1"<<endl;
   cout<<"B= "<<endl;
   cin>>b;
   cout<<"C= "<<endl;
   cin>>c;
   cout<<"D= "<<endl;
   cin>>d;

}

void cubik_equation::math()
{
```

```
   // Reduced equation: X^3 - 3pX - 2q = 0, where X = x-b/(3a)
   p = ( b * b - 3.0 * a * c ) / ( 9.0 * a * a );
   q = ( 9.0 * a * b * c - 27.0 * a * a * d - 2.0 * b * b * b ) / ( 54.0 * a * a * a );
   double offset = b / ( 3.0 * a );

   // Discriminant
   double discriminant = p * p * p - q * q;

   cout << "\nRoots:\n";
   if ( discriminant > 0 )          // set X = 2 sqrt(p) cos(theta) and compare 4 cos^3(theta)-3 cos(theta)
= cos(3 theta)
   {
      double theta = acos( q / ( p * sqrt( p ) ) );
      double r = 2.0 * sqrt( p );
      for ( int n = 0; n < 3; n++ ) cout << r * cos( ( theta + 2.0 * n * PII ) / 3.0 ) - offset << '\n';
   }
   else
   {
      double gamma1 = cbrt( q + sqrt( -discriminant ) );
      double gamma2 = cbrt( q - sqrt( -discriminant ) );

      cout << gamma1 + gamma2 - offset << '\n';

      double re = -0.5 * ( gamma1 + gamma2 ) - offset;
      double im = ( gamma1 - gamma2 ) * sqrt( 3.0 ) / 2.0;
      if ( discriminant == 0.0 )              // Equal roots (hmmm, floating point ...)
      {
         cout << re << '\n';
         cout << re << '\n';
      }
      else
      {
         cout << re << " + " << im << " * i\n";
         cout << re << " - " << im << " * i\n";
      }
   }
}
```

…… fournubers .cpp……..

```
#include "fournubers.h"

fournubers::fournubers()
{

}

void fournubers::get()
{
   cout<<"Enter a=";
   cin>>a;
   cout<<"Enter b=";
```

```
   cin>>b;
   cout<<"Enter c=";
   cin>>c;
   cout<<"Enter d=";
   cin>>d;
}

void fournubers::math()
{
  cout<<"\nA*B="<<a*b;
  cout<<"\tC*D="<<c*d<<endl;
}
```

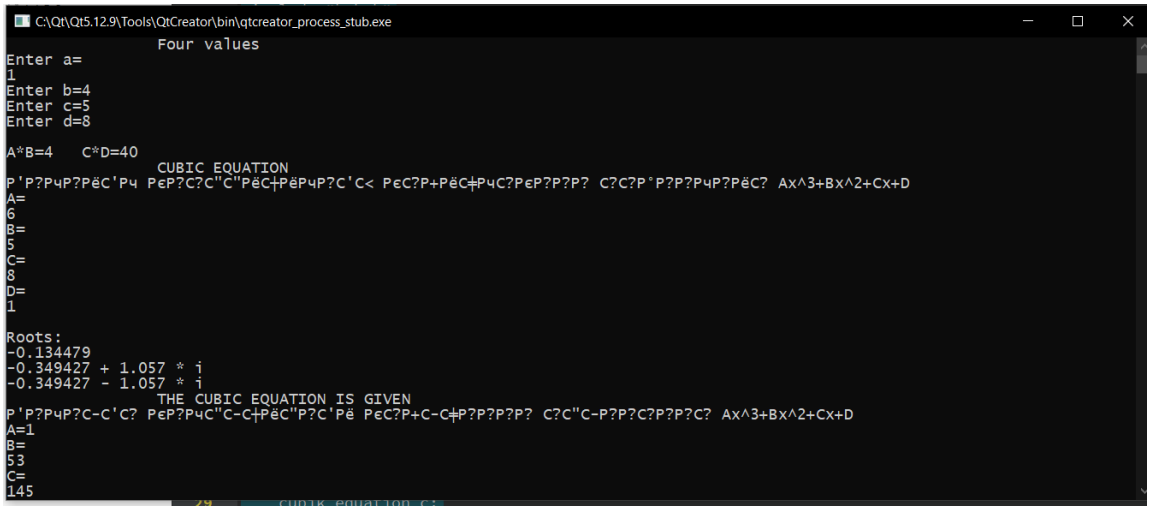
…… **kub** .h……

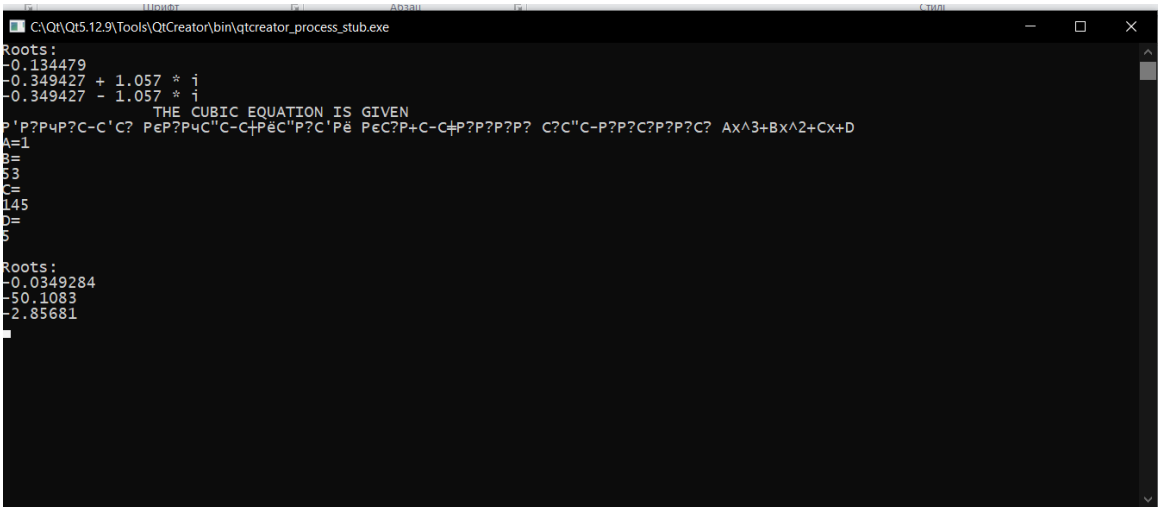
```
#include "kub.h"
#include <math.h>



kub::kub():fournubers(){}

kub::kub(double _A, double _B, double _C, double _D):fournubers(_A,_B,_C,_D){}

void kub::math()
{

     // Reduced equation: X^3 - 3pX - 2q = 0, where X = x-b/(3a)
      p = ( b * b - 3.0 * a * c ) / ( 9.0 * a * a );
      q = ( 9.0 * a * b * c - 27.0 * a * a * d - 2.0 * b * b * b ) / ( 54.0 * a * a * a );
     double offset = b / ( 3.0 * a );

     // Discriminant
     double discriminant = p * p * p - q * q;

     cout << "\nRoots:\n";
     if ( discriminant > 0 )         // set X = 2 sqrt(p) cos(theta) and compare 4 cos^3(theta)-3
cos(theta) = cos(3 theta)
     {
       double theta = acos( q / ( p * sqrt( p ) ) );
       double r = 2.0 * sqrt( p );
       for ( int n = 0; n < 3; n++ ) cout << r * cos( ( theta + 2.0 * n * PI ) / 3.0 ) - offset << '\n';
     }
     else
     {
       double gamma1 = cbrt( q + sqrt( -discriminant ) );
       double gamma2 = cbrt( q - sqrt( -discriminant ) );

       cout << gamma1 + gamma2 - offset << '\n';

       double re = -0.5 * ( gamma1 + gamma2 ) - offset;
       double im = ( gamma1 - gamma2 ) * sqrt( 3.0 ) / 2.0;
```

```
        if ( discriminant == 0.0 )              // Equal roots (hmmm, floating point ...)
        {
          cout << re << '\n';
          cout << re << '\n';
        }
        else
        {
          cout << re << " + " << im << " * i\n";
          cout << re << " - " << im << " * i\n";
        }
      }
}



void kub::get()
{
m:cout<<"Введите коэффициенты кубического уравнения Ax^3+Bx^2+Cx+D"<<endl;
cout<<"A= "<<endl;
cin>>a;
cout<<"B= "<<endl;
cin>>b;
cout<<"C= "<<endl;
cin>>c;
cout<<"D= "<<endl;
cin>>d;
if (a==0)
{
   cout<<"Ошибка a=0!"<<endl;
   goto m;
}
}
…… main .cpp……
#include <iostream>

#include "fournubers.h"
#include "kub.h"
#include "cubikequation.h"



using namespace std;

int main()
{

   setlocale(LC_ALL,"ukr");
   ////////////////////////////////////////////////////////////////
   cout<<"\t\tFour values"<<endl;
   fournubers f;
   f.get();
```

```
    f.math();

    ////////////////////////////////////////////////////////////////////
    cout<<"\t\tCUBIC EQUATION"<<endl;
    kub k;
    k.get();
    k.math();

    ////////////////////////////////////////////////////////////////////
    cout<<"\t\tTHE CUBIC EQUATION IS GIVEN"<<endl;
    cubik_equation c;
    c.get();
    c.math();


    return 0;
}
```

Результат





Висновок:

Ознайомитись з поняттям поліморфізму, його застосуванням та вивчив механізм його реалізації за допомогою віртуальних функцій