=================== Приклад поліморфізму ======================

// ConsoleApplication34.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

using namespace std;

class Base

{

public:

int a;

Base(int a) {

this->a = a;

}

virtual void show()

{

printf("Base a=%d\n",a);

}

};

class Derive :public Base

{

public:

int b;

Derive(int a, int b):Base(a) {

this->b = b;

}

void show()

{

printf("Derive a=%d, b=%d\n",a,b);

}

};

class Derive2 :public Base

{

public:

int c;

Derive2(int a, int c) :Base(a) {

this->c = c;

}

void show()

{

printf("Derive a=%d, c=%d\n", a, c);

}

};

int main()

{ Base\* b= new Base(5); //a=5

Base\*\* ff;

Derive\* d= new Derive(7,9); //a=7, b=9

Derive2\* d2 = new Derive2(89, 10); //a=89, c=10

b->show(); //b містить адресу об"єкта класу Base

b = d;

b->show(); //b містить адресу об"єкта класу Derive

b = d2;

b->show(); //b містить адресу об"єкта класу Derive2

printf("\n");

system("pause");

return 0;

}

Приклад.

#pragma once

#include <string>

using namespace std;

class Polygon

{

double\* sides;

int sidesCount;

void set\_sidesCount(int value);

public:

void set\_side(int index, double value);

double get\_side(int index);

int get\_sidesCount();

Polygon(int sidesCount);

Polygon(int sidesCount,double side1,...);

Polygon();

//-------- Методи ------

double Perimeter();

virtual double Area() = 0; //Чисто віртуальний метод

virtual string toString();

~Polygon();

};

------------------

#include "stdafx.h"

#include "Polygon.h"

void Polygon::set\_sidesCount(int value)

{

if (value<1)

{

throw "Err";

}

sidesCount = value;

sides = new double[sidesCount];

}

bool isIndexCorrect(int value,int sidesCount)

{

return value >= 0 && value < sidesCount;

}

void Polygon::set\_side(int index, double value)

{

if (!isIndexCorrect(index, sidesCount))

throw "index incorrect";

if (value < 0)

throw "Value must be >=0";

sides[index] = value;

}

double Polygon::get\_side(int index)

{

if (!isIndexCorrect(index, sidesCount))

throw "index incorrect";

return sides[index];

}

int Polygon::get\_sidesCount()

{

return sidesCount;

}

Polygon::Polygon(int sidesCount)

{

set\_sidesCount(sidesCount);

for (int i = 0; i < sidesCount; i++)

{

sides[i] = 0;

}

}

Polygon::Polygon(int sidesCount, double side1, ...):Polygon(sidesCount)

{

double\* temp = &side1;

for (int i = 0; i < sidesCount; i++)

{

sides[i] = temp[i];

}

}

Polygon::Polygon():Polygon(10)

{

}

double Polygon::Perimeter()

{

double s = 0;

for (int i = 0; i < sidesCount; i++)

{

s += sides[i];

}

return s;

}

string Polygon::toString()

{

string res = to\_string(sides[0]);

for (int i = 1; i < sidesCount; i++)

{

res += ", " + to\_string(sides[i]);

}

return res;

}

Polygon::~Polygon()

{

delete[]sides;

}

--------------------------------------------------------------

#pragma once

#include "Polygon.h"

class Rectangle :

public Polygon

{

public:

using Polygon::Perimeter;

void set\_a(double value);

double get\_a();

void set\_b(double value);

double get\_b();

Rectangle(double a, double b);

Rectangle();

double Area();

string toString();

~Rectangle();

};

------------------------------------------------

#include "stdafx.h"

#include "Rectangle.h"

void Rectangle::set\_a(double value)

{

set\_side(0, value);

}

double Rectangle::get\_a()

{

return get\_side(0);

}

void Rectangle::set\_b(double value)

{

set\_side(1,value);

}

double Rectangle::get\_b()

{

return get\_side(1);

}

Rectangle::Rectangle(double a, double b):Polygon(2,a,b)

{

}

Rectangle::Rectangle():Rectangle(0,0)

{

}

double Rectangle::Area()

{

return get\_a()\*get\_b();

}

string Rectangle::toString()

{

return "Rectangle : "+Polygon::toString();

}

Rectangle::~Rectangle()

{

}

-------------------------------------------------------------------------------------

#pragma once

#include "Polygon.h"

class Triangle :

public Polygon

{

public:

using Polygon::Perimeter;

void set\_side(char sideName, double value);

double get\_side(char sideName);

Triangle(double a, double b, double c);

Triangle();

string toString();

double Area();

~Triangle();

};

----------------------------

#include "stdafx.h"

#include "Triangle.h"

#include <Math.h>

void Triangle::set\_side(char sideName, double value)

{

//int index = towupper(sideName) - 'A';

//set\_side(index, value);

switch (towupper(sideName))

{

case 'A':set\_side(0, value);

break;

case 'B':set\_side(1, value);

break;

case 'C':set\_side(2, value);

break;

default:

throw "Name incorrect";

}

}

double Triangle::get\_side(char sideName)

{

//int index = towupper(sideName) - 'A';

//return get\_side(index);

switch (towupper(sideName))

{

case 'A':return Polygon::get\_side(0);

case 'B':return Polygon::get\_side(1);

case 'C':return Polygon::get\_side(2);

default:

throw "Name incorrect";

}

}

Triangle::Triangle(double a, double b, double c):Polygon(3,a,b,c)

{

}

Triangle::Triangle():Triangle(0,0,0)

{

}

string Triangle::toString()

{

return "Triangle : " + Polygon::toString();

}

double Triangle::Area()

{

double p = Perimeter() / 2;

return sqrt(p\*(p-get\_side('A'))\*(p - get\_side('B'))\*(p - get\_side('C')));

}

Triangle::~Triangle()

{

}

================================= main ===============

// ConsoleApplication34.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

#include "Polygon.h"

#include "Rectangle.h"

#include "Triangle.h"

using namespace std;

int main()

{

Polygon\* \* figures = new Polygon\*[5];

figures[0] = new Triangle(2, 2, 1);

figures[1] = new Rectangle(6, 8);

figures[2] = new Triangle(3, 2, 4);

figures[3] = new Rectangle(6, 8);

figures[4] = new Rectangle(9, 8);

for (int i = 0; i < 5; i++)

{

printf("%s \n",figures[i]->toString().data());

}

printf("--------------------\n");

double sum = 0;

for (int i = 0; i < 5; i++)

{

sum += figures[i]->Area();

}

printf("Sum=%f\n",sum);

system("pause");

return 0;

}