***ЛАБОРАТОРНА РОБОТА №4***

**Тема:** Успадковування класів  
**Мета:** Ознайомитись зі способами та механізмами успадкування класів та навчитись використовувати їх для побудови об’єктно-орієнтованих програм.

***ПОРЯДОК ВИКОНАННЯ РОБОТИ***

**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().

Код програми:

#include <QCoreApplication>

#include <string.h>

#include <iostream>

using namespace std;

class Publication

{

string name;

float price;

public:

void getData(){ cout << "Name: " << name << endl << "Price: " << price << endl;}

void putData() {cout << "Write name: "; cin >> name;

cout << "Write price: "; cin >> price; }

};

class Book: Publication

{

int pageAmount;

public:

void getData(){ Publication::getData(); cout << "Page amount: " << pageAmount << endl; }

void putData() {Publication::putData(); cout << "Write page amount: "; cin >> pageAmount; }

};

class Type: Publication

{

float writeTime;

public:

void getData(){ Publication::getData(); cout << "Write time: " << writeTime << endl; }

void putData() {Publication::putData(); cout << "Write writing time: "; cin >> writeTime; }

};

int main(int argc, char \*argv[])

{

QCoreApplication a(argc, argv);

Book b;

Type t;

b.putData();

t.putData();

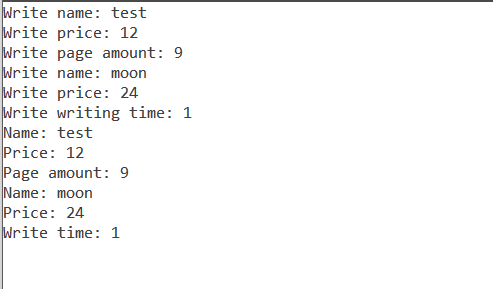
b.getData();

t.getData();

return a.exec();

}

Вивід програми:



**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:

#include <QCoreApplication>

#include <iostream>

#include <math.h>

using namespace std;

class Point

{

float x,y;

public:

Point(float \_x, float \_y){x = \_x; y = \_y;}

Point():Point(0,0){}

float GetX(){return x;}

float GetY(){return y;}

void SetX(float \_x){ x = \_x;}

void SetY(float \_y){ x = \_y;}

string Get(){return "(" + to\_string(x) + ";" + to\_string(y) + ")";}

float GetLength(){return (float)sqrt(pow(x,2)+pow(y,2));}

float GetAngle(){return atan2(y,x);}

float DistanceBetween(Point p)

{

return (float)sqrt(pow(x-p.x,2)+pow(y-p.y,2));

}

Point& operator-(Point& p)

{

return \*new Point(x-p.x, y-p.y);

}

};

class Fourangle

{

Point verticles[4];

public:

Fourangle():Fourangle(\*new Point(0,0), \*new Point(0,0), \*new Point(0,0), \*new Point(0,0)){}

Fourangle(Point &v1, Point &v2, Point &v3, Point &v4)

{

verticles[0] = v1;

verticles[1] = v2;

verticles[2] = v3;

verticles[3] = v4;

}

~Fourangle() {}

void SetVerticle(int pos, Point &p)

{

verticles[pos] = p;

}

Point& GetVerticle(int pos)

{

return verticles[pos];

}

void Show()

{

cout << "Rectangle with verticles: ";

for(int i = 0; i < 4; i++)

cout << verticles[i].Get() << " ";

cout << endl;

}

void Input()

{

int x,y;

for(int i = 0; i < 4; i++)

{

cout << "Corner " << i+1 << ": ";

cin >> x >> y;

verticles[i].SetX(x);

verticles[i].SetY(y);

}

}

static float Dobutok(float a, float b)

{

return a\*b;

}

};

class Paralelogram : Fourangle

{

private:

float GetCornerAngle()

{

return abs((GetVerticle(1) - GetVerticle(0)).GetAngle() - (GetVerticle(3) - GetVerticle(0)).GetAngle());

}

public:

Paralelogram():Fourangle(){}

Paralelogram(Point &v1, Point &v2, Point &v3, Point &v4):Fourangle(v1, v2, v3, v4){}

~Paralelogram(){}

float Square()

{

return Dobutok(GetVerticle(0).DistanceBetween(GetVerticle(1)), GetVerticle(1).DistanceBetween(GetVerticle(2)))\*sin(GetCornerAngle());

}

};

int main(int argc, char \*argv[])

{

QCoreApplication a(argc, argv);

return a.exec();

}

Висновок: Ознайомився зі способами та механізмами успадкування класів та навчився використовувати їх для побудови об’єктно-орієнтованих програм.