Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра “Електронних обчислювальних машин”



**Звіт з лабораторної роботи №7**

на тему:

“Множинне спадкування. Поліморфізм”

**Виконав:**

Ст. гр. КІ-15

Стецик О.І.

**Перевірив:**

Викладач

Козак Н.Б.

Львів – 2020

**Мета роботи:** познайомитися із множинним спадкуванням класів та поліморфізмом.

***Завдання:***

Спроектувати і реалізувати ієрархію класів, що описують предметну область згідно варіанту, яка реалізується класом 1. Клас 1 в свою чергу утворюється шляхом множинного спадкування класів 2 і 3 кожен з яких в свою чергу успадковує клас 4. Додаткові вимоги:

1. Базовий клас містить мінімум один віртуальний метод, один невіртуальний метод і одну динамічно створювану властивість.

2. Забезпечити механізми коректної роботи конструкторів і деструкторів.

3. Перевантажити оператор присвоєня з метою його коректної роботи.

4. Кожен з класів має містити мінімум одину властивість і 4 методи.

5. Написати main() функцію де створити об‘єкт класу 1 і продемонструвати різницю між статичним і динамічним поліморфізмом.



******

*Код програми:*

#include <iostream>

#include <conio.h>

#include "Header.h"

using namespace std;

int main()

{

CDevice cd;

CClock cc;

CWasher cw;

CCW ccw;

cd.set\_name("LG");

cd.set\_price(10000);

cd.show\_info();

cout << endl;

cw.set\_efficiency(65);

cw.set\_drying(false);

cw.show\_info();

cout << endl;

cc.set\_signal(true);

cc.set\_max\_time(2);

cc.show\_info();

cout << endl;

ccw.set\_func\_num(125);

ccw.show\_info();

cout << endl;

system("pause");

return 0;

}

**Header:**

#pragma once

#include <iostream>

using namespace std;

class CDevice {

private:

double weight;

double price;

char company\_name[30];

public:

CDevice()

{

this->weight = 50;

this->price = 7000;

}

CDevice(double Weight, double Price)

{

this->weight = Weight;

this->price = Price;

}

CDevice& operator= (const CDevice& drob)

{

weight = drob.weight;

price = drob.price;

return \*this;

}

void set\_name(const char\* name)

{

strcpy\_s(company\_name, name);

}

void set\_price(double Price)

{

this->price = Price;

}

void set\_weight(double Weight)

{

this->weight = Weight;

}

virtual void show\_info()

{

cout << "It's weight is: " << this->weight << " kg" << endl;

cout << "It's price is: " << this->price << " gryvnas" << endl;

cout << "It's company name is: " << this->company\_name << endl;

}

virtual ~CDevice() {};

};

class CWasher : CDevice {

private:

int efficiency;

bool drying;

public:

CWasher()

{

this->efficiency = 70;

std::cout << std::boolalpha;

this->drying = true;

}

CWasher(int Efficiency, bool Drying)

{

this->efficiency = Efficiency;

std::cout << std::boolalpha;

this->drying = Drying;

}

void set\_efficiency(int eff)

{

this->efficiency = eff;

}

void set\_drying(bool test)

{

this->drying = test;

}

CWasher& operator= (const CWasher& drob)

{

efficiency = drob.efficiency;

drying = drob.drying;

return \*this;

}

void show\_info()

{

cout << "Efficiency is: " << this->efficiency << " %" << endl;

cout << "Drying availability: " << this->drying;

}

virtual ~CWasher() {};

};

class CClock : CDevice {

private:

double max\_time;

bool signal;

public:

CClock()

{

this->signal = false;

this->max\_time = 1;

}

CClock(double Max\_time, bool Signal)

{

this->max\_time = Max\_time;

this->signal = Signal;

}

CClock& operator= (const CClock& drob)

{

signal = drob.signal;

max\_time = drob.max\_time;

return \*this;

}

void set\_max\_time(double Max\_time)

{

this->max\_time = Max\_time;

}

void set\_signal(bool Signal)

{

this->signal = Signal;

}

void show\_info()

{

cout << "Signal availability : " << this->signal << endl;

cout << "Maximal time is: " << this->max\_time <<" hours"<< endl;

}

virtual ~CClock() {};

};

class CCW : CClock, CWasher {

private:

int func\_number;

public:

CCW()

{

this->func\_number = 0;

};

CCW(int Func\_num)

{

this->func\_number = Func\_num;

};

CCW& operator= (const CCW& drob)

{

func\_number = drob.func\_number;

return \*this;

}

void set\_func\_num(int Func\_num)

{

this->func\_number = Func\_num;

}

void show\_info()

{

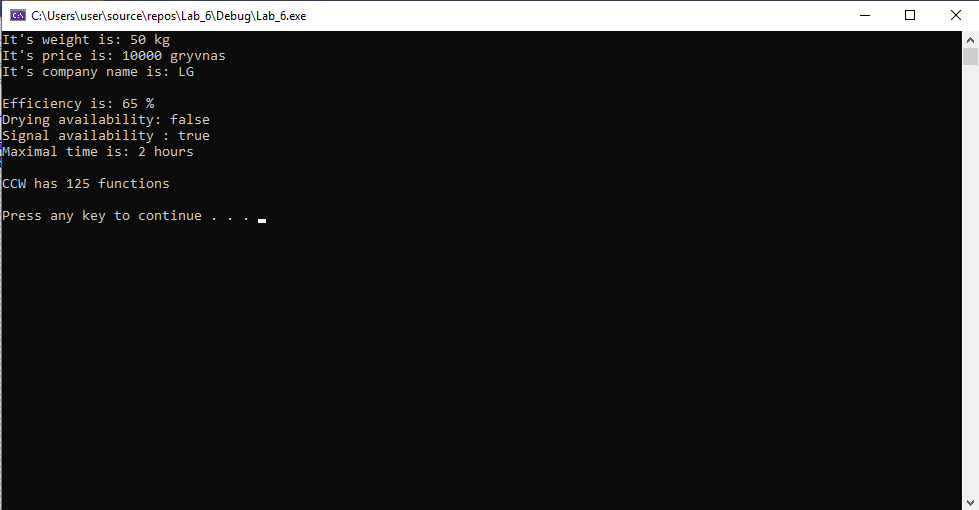
cout << "CCW has " << func\_number << " functions" << endl;

}

virtual ~CCW() {};

};

*Вікно результату:*



*Висновок:* я познайомився з множинним спадкуванням класів та з поліморфізмом.