Міністерство науки і освіти України

коломийський політехнічний коледж

національного університету «Львівська політехніка»

**Циклова комісія: Інженерія програмного забезпечення**

**Практична робота №4**

**З дисципліни**

**«Об’єктно-орієнтоване програмування»**

Виконав студент групи П-31

Мотрунич Владислав Іванович

Коломия 2018

Теорія

Вказівник – це змінна, яка містить адресу місця пам’яті де знаходиться об’єкт.

Вказівник це дуже потужний інструмент за допомогою якого можна реалізовувати шаблони різного рівня. Також ними можна реалізовувати звертання до масивів динамічно.

Для більш ефективного використання пам’яті часто доводиться розроблювати класи, в яких розмір об’єкту залежить від даних, що в ньому зберігаються. В таких структурах часто об’єм потрібної пам’яті стає відомим лише безпосередньо перед ініціалізацією конкретного об’єкта даного класу. Це призводить до необхідності динамічного керування пам’яттю.

Задача

1. Створити однозв’язний циклічний список і вивести результат.

2. Створити клас монстрів, які будуть атакувати героя. Створити купу монстрів за допомогою масиву і вказівників.

Код програми 1.

#include "pch.h"

#include <iostream>

#include <conio.h>

#include <cstdlib>

using namespace std;

class List

{

public:

int Value;

List \*next;

List \*NewChain(int val)

{

List \*l = new List;

l->next = NULL;

l->Value = val;

return l;

}

};

static class Controller

{

public:

static List\* makeList(List \*ch,int v)

{

cout << "enter values of list. enter -1 to stop" << endl;

cin >> v;

ch->Value = v;

List \*ph = ch;

cin >> v;

do

{

ch->next = ch->NewChain(v);

ch = ch->next;

cin >> v;

} while (v != -1);

ch->next = ph;

return ch;

}

};

static class View

{

public:

static void show(List\*ph, int v)

{

cout << "output.next:1 stop:0" << endl;

cin >> v;

do

{

if (v == 1)cout << "next thing = " << ph->next->Value << endl;

ph = ph->next;

cin >> v;

} while (v != 0);

}

};

int main()

{

int v = 0;

List \*ch = new List;

ch = Controller::makeList(ch, v);

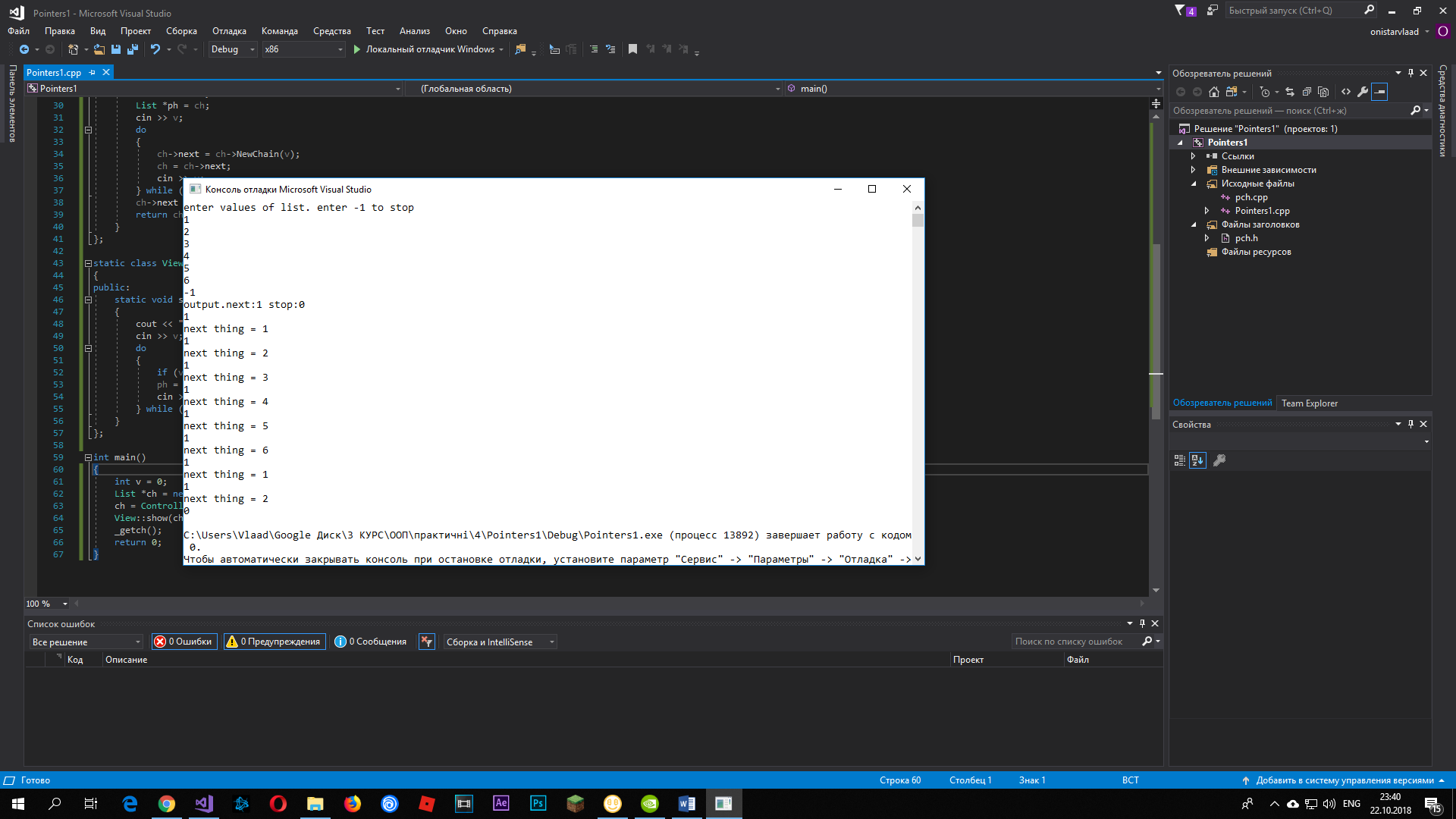
View::show(ch, v);

\_getch();

return 0;

}

Консоль 1



Код програми 2.

#include "pch.h"

#include <iostream>

#include <cstdlib>

#include <conio.h>

#include <Windows.h>

using namespace std;

class Hero

{

int attak = 20;

public:

int hp = 100;

int getAtk()

{

return attak;

}

int getHp()

{

return hp;

}

void getDmg()

{

cout << "oiii" << endl;

}

void Die()

{

if (this->hp <= 0)cout << "oh no, monsters killed me. RIP" << endl;

}

};

class Monster

{

int hp = 100;

int attak = 20;

public:

int getAtk()

{

return attak;

}

int getHp()

{

return hp;

}

void toAttak(Hero &obj)

{

obj.hp -= this->attak;

}

};

int main()

{

Hero Gervant;

Monster \*goblins = new Monster[5];

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

{

(goblins + i)->toAttak(Gervant);

Gervant.getDmg();

Sleep(1000);

}

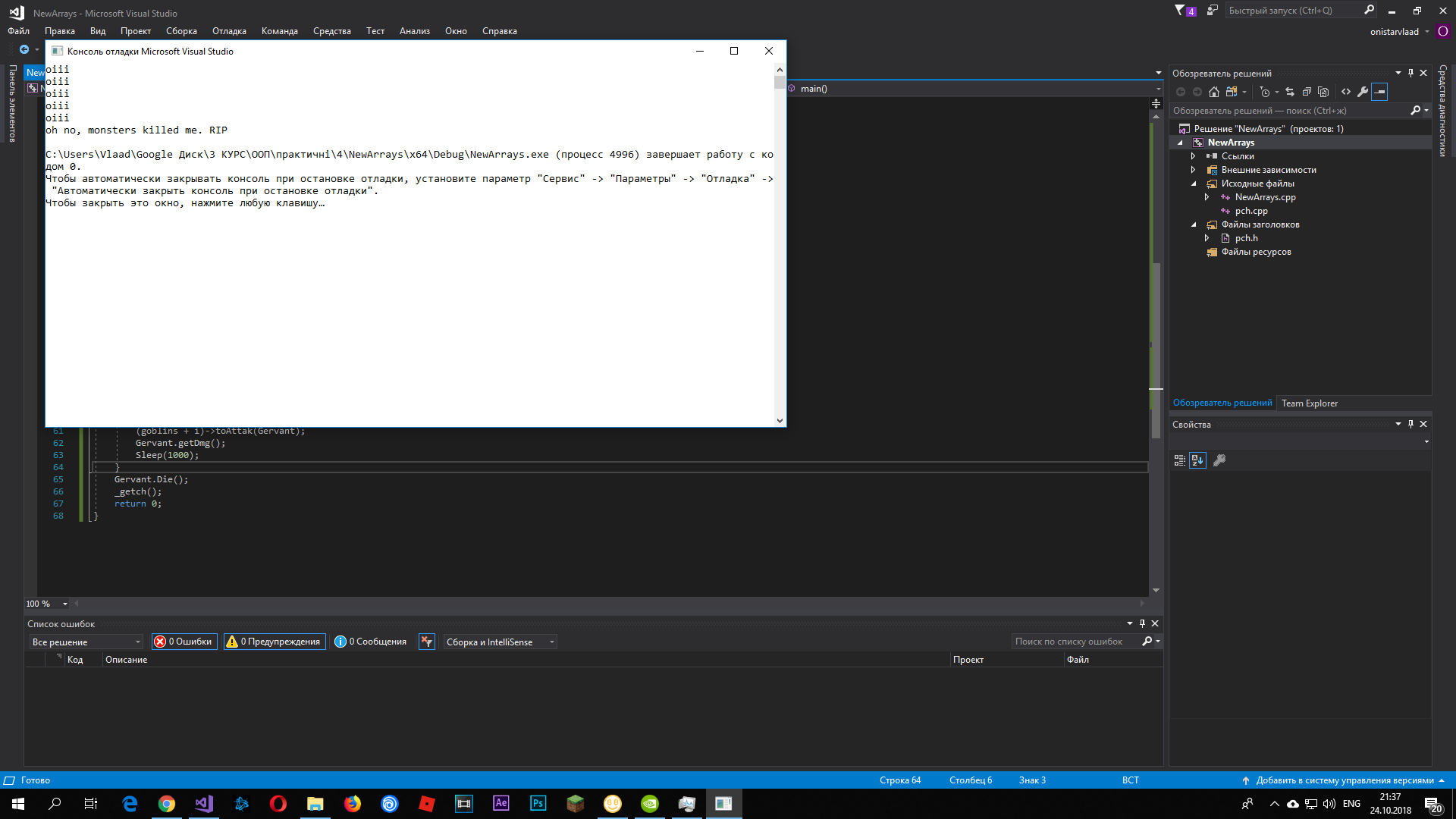
Gervant.Die();

\_getch();

return 0;

}

Консоль 2



Висновок

За допомогою даної практичної роботи я оволодів вказівниками в ООП парадигмі. За допомогою них я реалізував таку структуру, як список і показав, як можна використовувати вказівники для керування масивами об’єктів.