Міністерство освіти науки і України

Західноукраїнський національний університет

Факультет компʼютерних інформаційних технологій

Кафедра інформаційних-обчислювальних систем і управління

Лабораторна робота №6

Виконала:

Студентка КН-12

Мисліцька Вероніка

Тернопіль 2024

**Тема:** Реалізація принципу поліморфізму в ООП. Перевантаження операцій.

**Мета робота:** засвоїти принципи перевантаження операцій, навчитися створювати ієрархії класів та використовувати поліморфізм.

**Варіант 15**

**Хід роботи:**

**Завдання 1.**

1. Код програми.

#include <iostream>

**using** **namespace** std;

**class** Point {

**private**:

**float** x, y;

**public**:

Point(**float** x = 0, **float** y = 0) : x(x), y(y) {}

**float** getX() **const** { **return** x; }

**float** getY() **const** { **return** y; }

};

**class** Circle {

**private**:

Point center;

**float** radius;

**public**:

Circle(**float** x = 0, **float** y = 0, **float** radius = 1) : center(x, y), radius(radius) {}

**float** getRadius() **const** { **return** radius; }

Point getCenter() **const** { **return** center; }

};

**int** main() {

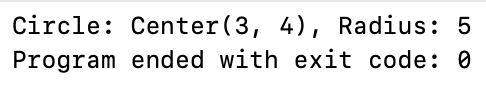
Circle c(3, 4, 5);

cout << "Circle: Center(" << c.getCenter().getX() << ", " << c.getCenter().getY() << "), Radius: " << c.getRadius() << endl;

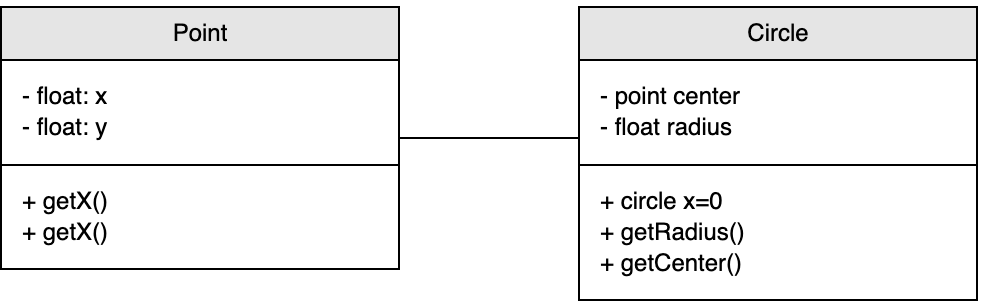
**return** 0;

}

1. Вивід програми.



1. Діаграма.



**Завдання 2**

1. Код програми.

#include <iostream>

**using** **namespace** std;

**class** Integer {

**private**:

**int** value;

**public**:

Integer() : value(0) {}

Integer(**int** val) : value(val) {}

Integer(**const** Integer &other) : value(other.value) {}

Integer& **operator**=(**const** Integer &other) {

**if** (**this** != &other) {

value = other.value;

}

**return** \***this**;

}

**bool** **operator**==(**const** Integer &other) **const** {

**return** value == other.value;

}

**bool** **operator**!=(**const** Integer &other) **const** {

**return** !(\***this** == other);

}

Integer& **operator**++() {

++value;

**return** \***this**;

}

Integer **operator**++(**int**) {

Integer temp(\***this**);

++(\***this**);

**return** temp;

}

Integer& **operator**--() {

--value;

**return** \***this**;

}

Integer **operator**--(**int**) {

Integer temp(\***this**);

--(\***this**);

**return** temp;

}

Integer **operator**+(**const** Integer &other) **const** {

**return** Integer(value + other.value);

}

Integer **operator**-(**const** Integer &other) **const** {

**return** Integer(value - other.value);

}

**bool** **operator**&&(**const** Integer &other) **const** {

**return** value && other.value;

}

**bool** **operator**||(**const** Integer &other) **const** {

**return** value || other.value;

}

**bool** **operator**!() **const** {

**return** !value;

}

**friend** ostream& **operator**<<(ostream &out, **const** Integer &integer) {

out << integer.value;

**return** out;

}

};

**int** main() {

Integer a(5);

Integer b(3);

Integer c = a + b;

cout << "a + b = " << c << endl;

Integer d = a - b;

cout << "a - b = " << d << endl;

Integer e = ++a;

cout << "++a = " << e << endl;

Integer f = b++;

cout << "b++ = " << f << endl;

Integer g = --a;

cout << "--a = " << g << endl;

Integer h = b--;

cout << "b-- = " << h << endl;

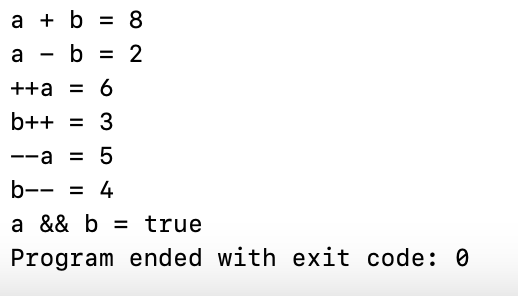
**bool** result = a && b;

cout << "a && b = " << (result ? "true" : "false") << endl;

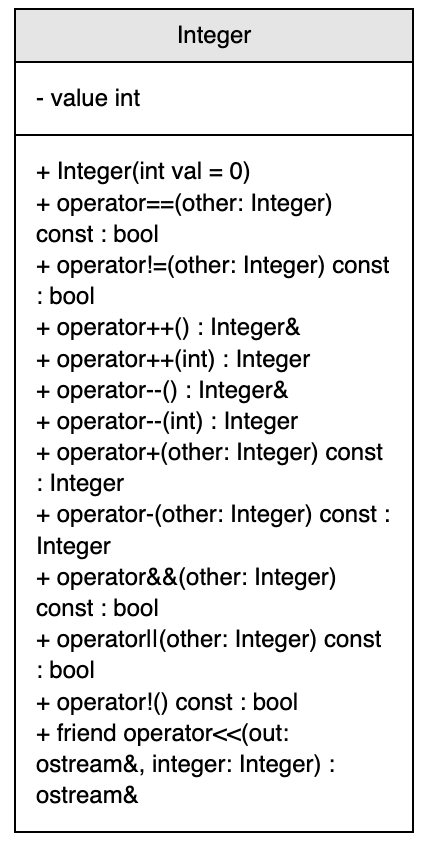
**return** 0;

}

1. Вивід програми.



1. Діаграма.



**Висновок:** Сьогодні я засвоїла принципи перевантаження операцій, створювати ієрархії класів та використовувати поліморфізм.