Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського” Факультет

інформатики та обчислювальної техніки Кафедра

інформатики та програмної інженерії

Звіт

З лабораторної роботи №4 з дисципліни

«Основи програмування 2.

Модульне програмування»

«Перевантаження операторів»

Варіант 27

Виконав студент ІП-11 Савенко Олексій Андрійович

(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська І.І.

(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота 4

Перевантаження операторів

Мета роботи – вивчити механізми створення класів з використанням перевантажених операторів(операцій).

Варіант 27

27. Визначити клас "Точка", який задається координатами точки на площині. Реалізувати для нього декілька конструкторів, геттери, метод визначення квадранта системи координат (його номера), в якому знаходиться дана точка. Перевантажити оператори: префіксний "++" – для збільшення x-координати точки на 1, постфіксний "++" – для збільшення y–координати точки на 1, "-" – для визначення відстані між двома точками. Створити три точки (T1, T2, T3), використовуючи різні конструктори. Інкрементувати x-координату точки T1 і y–координату точки T2. Визначити відстань між отриманими точками T1 і T2. З'ясувати, якому квадранту належить точка T3.

Постановка завдання

Розробити клас, який представляє собою точку, яка має координати X та Y,реалізувати декілька конструкторів, гетери, сетери, метод визначення квадранта системи координат(його номера в якому знаходиться ця точка).Використати на практиці перевантаження операторів – префіксний інкремент для збільшення x-координати точки на 1, постфіксний для збільшення y-координати на 1, - для визначення відстані між двома точками. Створити три точки за допомогою різних конструкторів, інкрементувати x-координату точки T1 і y-координату точки T2, визначити відстань між ними. З'ясувати, якому квадранту належить точка T3.

Програмний код на мові C++:

Lab4Op++.cpp

```
#include <iostream>
#include "Point.h"


int main() {

	cout << "Enter X of T1 Point: ";
	double X;
	cin >> X;
	cout << "Enter Y of T1 Point: ";
	double Y;
	cin >> Y;

	cout << "Enter quadrant of T2 Point: ";
	int quadrant;
	cin >> quadrant;
	//Створення об'єктів класу точки за допомогою різних конструкторів
	Point T1(X, Y), T2(quadrant), T3;
	Point points[]{ T1,T2,T3 };
	cout << "----------------------------------------"<<endl;
	cout << "Points before changing"<<endl;
	Show(points);   //Виведення точок перед зміною за допомогою операторів
	//Використання перевантажених операторів інкременту
	++points[0];
	points[1]++;
	cout << "----------------------------------------" << endl;
	cout << "Points after changing" << endl;
	Show(points); //Виведення після зміни
	cout << "----------------------------------------" << endl;
	//Використання перевантаженого оператору віднімання для знаходження відстані
	cout << "The distance between T1 and T2 points is " << points[0] - points[1]<<endl;

	cout << "Quadrant of point T3 is " << T3.GetQuadrant() << endl;


}
```

Point.h

```
#pragma once
#include <iostream>
using namespace std;

class Point {
	//Атрибути класа точки за осями
private:
	double X;
	double Y;

public:
	//Гетери та Сетери
	void SetX(double);


	double GetX();

	void SetY(double);

	double GetY();

	int GetQuadrant();
	//Конструктор за замовчуванням
	Point();
	//З параметрами
	Point(double, double);
	//Конструктор за чвертю
	Point(int);


	//Функції перевантаження операторів
	friend double operator - (Point, Point);
	Point& operator++();
	Point operator++(int);

};


void Show(Point[]); //Виведення масиву трьох точок
```

Point.cpp

```
#include "Point.h"
#include <cmath>


//Гетери і сетери відповідних атрибутів класу точки
void Point::SetX(double X) {
	this->X = X;
}

double Point::GetX() {
```

```
	return this->X;
}

void Point::SetY(double Y) {
	this->Y = Y;
}

double Point::GetY() {
	return Y;
}


//Функція для знаходження квадранта відповідної точки
int Point::GetQuadrant() {
	if (X > 0 && Y > 0) return 1;
	else if (X < 0 && Y > 0) return 2;
	else if (X < 0 && Y < 0) return 3;
	else if (X > 0 && Y < 0) return 4;
	else return 0;

}
//Конструктор з введенням даних
Point::Point() {
	cout << "Enter T3 Point x coordinate: ";
	cin >> this->X;
	cout << "Enter T3 Point y coordinate: ";
	cin >> this->Y;
}
//Конструктор з передаванням параметрів для точок та ініціалізації атрибутів об'єкта з допомогою Сетерів
Point::Point(double X,double Y) {
	SetX(X);
	SetY(Y);
}



//Рандомна генерація точки в залежності від введеного квадранта користувачем
Point::Point(int quadrant) {

	srand(time(NULL));
	switch (quadrant) {
	case 1:
		SetX(round((double)rand() / RAND_MAX * 10 * 10) / 10);
		SetY(round((double)rand() / RAND_MAX * 10 * 10) / 10);
		break;
	case 2:
		SetX(round(((double)rand() / RAND_MAX * 10 - 10) * 10) / 10);
		SetY(round((double)rand() / RAND_MAX * 10 * 10) / 10);
		break;
	case 3:
		SetX(round(((double)rand() / RAND_MAX * 10 - 10) * 10) / 10);
		SetY(round(((double)rand() / RAND_MAX * 10 - 10) * 10) / 10);
		break;
	case 4:
		SetX(round((double)rand() / RAND_MAX * 10 * 10) / 10);
		SetY(round(((double)rand() / RAND_MAX * 10 - 10) * 10) / 10);
		break;
	default:
		cout << "Incorrect quadrant in constructor!\nSetted an default zero values"<<endl;
		SetX(0);
```

```
                SetY(0);
        }
        }

//Перевантаження оператора - для знайдення відстанні між двома точками
double operator - (Point T1, Point T2) {

        return sqrt(pow(T2.GetX()-T1.GetX(),2)+pow((T2.GetY()-T1.GetY()),2));

}
//Перевантаження префіксного інкремента для збільшення X на 1
Point& Point::operator++() {
        ++this->X;
        return (*this);
}

Point Point::operator++(int) {
        return{ this->X, this->Y++ };
}
// Виведення масиву у якому зберігаються об'єкти класу точок
void Show(Point points[]) {
        for (int i = 0; i < 3; i++) {
                cout << "Point T" << i + 1 << " X:" << points[i].GetX() << " Y:" << points[i].GetY() << endl;
        }

}
```

Висновок

Отже, я вивчив механізми створення класів з використанням перевантажених операторів(операцій). Я виконав завдання застосувавши та закріпивши на практиці набуті теоретичні відомості щодо перевантаження методів, операторів, інкапсуляції, конструкторів класу. Мною був розроблений клас Точки який має два поля координату **X** та **Y**, декілька конструкторів, перевантаження операторів для більш спрощеної, зрозумілої взаємодії з об’єктами класу та їх полями, а також метод **GetQuadrant**, який обробляє поля відповідного об’єкта класу та повертає значення яке дорівнює квадранту, у якому лежить дана точка. У підсумку моє завдання було виконано, а набуті знання закріплені, також було проведено тестування програми для знаходження помилок у ній та їх виправлення.