[lab09 보고서]

자율전공학과

2019111677

김지연

1. 소스코드

#include <iostream>

using namespace std;

class Complex {

private:

double re, im; //re은 정수 im은 허수

public:

friend ostream& operator<<(Complex& c, ostream& os); //프렌드함수정의(클래스 내부 데이터에 쉽게 접근할 수 있게 하기 위함) 출력스트림연산자 전역함수로 정의

friend ostream& operator<<(ostream& os, Complex& c); //프렌드함수정의. 출력스트림연산자 전역함수로 정의

friend Complex operator+(Complex& c3, int a); //프렌드함수정의. +연산자 정의

Complex(double re, double im) {

this->re = re;

this->im = im;

}

Complex operator+(Complex& c2)const //c1+c2 가 계산되는 함수. 연산자 중복(멤버함수로 정의)

{

Complex c(0.0, 0.0); //값을 초기화

c.re = this->re + c2.re; //this->re가 호출의 주체 뒤에서 c1으로 출력. c1과 c2의 정수부분 계산

c.im = this->im + c2.im; //this->imdl 호출의 주체 뒤에서 c1으로 출력. c1과 c2의 허수부분 계산

return c; //값 반환하기

}

};

Complex operator+(Complex& c3, int a) { //c3+1 객체와 정수가 계산되는 함수. 연산주 중복(비멤버 함수로 정의)

Complex c(0.0, 0.0); //값을 초기화

c3.re = c3.re + a; //c3의 정수 부분과 정수형 a의 값만 계산

return c3; //값 반환

}

ostream& operator <<(ostream& os, Complex& c) //cout<<c1. 연산자 중복(비멤버 함수로 정의)

{

os << c.re << "+" << c.im << "i" << endl; //정수부분과 허수부분을 차례로 출력

return os; //스트림 객체를 반환

}

ostream& operator <<(Complex& c, ostream& os) //c1<<cout. 연산자 중복(비멤버 함수로 정의). 위의 함수와 다른점은 출력하는 순서

{

os << c.re << "+" << c.im << "i" << endl; //정수부분과 허수부분 차례로 출력

return os; //스트림 객체를 반환

}

int main(void)

{

Complex c1(1, 2), c2(3, 4);

Complex c3 = c1 + c2; //c1.operator+(c2)

c1 << cout; //operator+(c1, cout) 전역함수의 순서를 바꾸는 것으로 해결

cout << c3 << endl; // operator<<(cout,c3) <<연산자를 이용하여 객체 출력

c3 = c3 + 1; // operator+(c3, a) 객체와 정수가 계산되야함(a는 정수형)

cout << c3 << endl; // operator<<(cout,c3) <<연산자를 이용하여 객체 출력

cout << "c1, c2, c3 : " << endl; //문자열 출력

c3 << (c2 << (c1 << cout)); //묶어져 있는 순서대로 출력

cout << endl;

cout << "c1, c2, c3 : " << endl; //문자열 출력

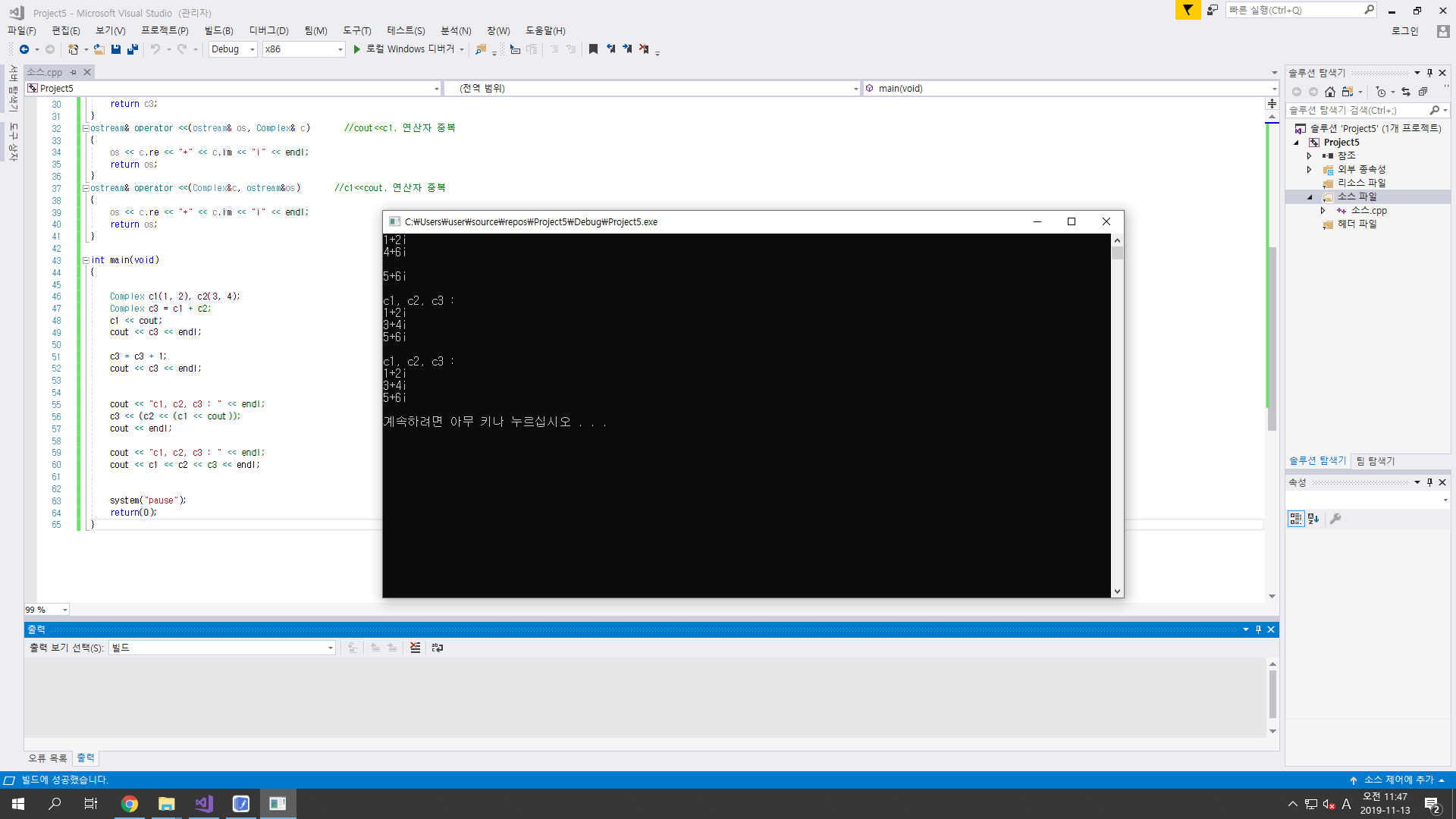
cout << c1 << c2 << c3 << endl; //operator<<(cout,c1,c2,c3) <<연산자를 이용하여 객체 출력

system("pause");

return(0);

}

2. 실행결과화면



3. 문제 정의 및 분석

메인함수에 객체들을 출력하고 연산할 수 있는 연산자가 없음

호출할 수 있는 함수도 없음

연산자 중복정의를 통해 문제 해결

복소수 클래스의 오버로딩

* 메인함수에 필요한 연산자들은 +연산자와 <<출력스트림 연산자
* c1과c2를 계산하기 위해 멤버함수로 함수 만들고 c3로 함수 호출
* c3+1은 c3의 정수부분과 뒤에 나오는 정수만 계산되야 함
* c1<<cout은 전역함수를 만들 때 순서를 바꿔주어서 해결

[필요한 개념]

프렌드 함수: 내부 데이터에 쉽게 접근할 수 있게 해주는 외부 함수

연산자 중복: operator. 정수나 실수를 넘어서 클래스 타입도 계산할 수 있도록 함

외부함수(비멤버 함수)로 연산자 구현

v1+v2

-> Operator+(v1,v2)

멤버 함수로 연산자 구현

v1+v2

-> v1.operator+(v2)