객체지향프로그래밍  
HW07 Operator Overloading

C211123 이준선

2023.05.07

# 코드

#include <iostream>

class Complex

{

private:

double real;

double imaginary;

public:

Complex(double \_real) : real(\_real), imaginary(0)

{

}

Complex(double \_real = 0, double \_imaginary = 0) : real(\_real), imaginary(\_imaginary)

{

}

void Output()

{

std::cout << real << " + " << imaginary << "i" << std::endl;

}

Complex &operator+=(const Complex &);

Complex &operator-();

friend Complex operator+(const Complex &, const Complex &);

friend Complex operator++(Complex &);

friend Complex operator++(Complex &, int);

friend std::ostream &operator<<(std::ostream &os, const Complex &);

};

Complex &Complex::operator+=(const Complex &rhs)

{

real += rhs.real;

imaginary += rhs.imaginary;

return \*this;

}

Complex &Complex::operator-()

{

real = -real;

imaginary = -imaginary;

return \*this;

}

Complex operator+(const Complex &lhs, const Complex &rhs)

{

Complex temp(lhs);

temp += rhs;

return temp;

}

Complex operator++(Complex &rhs)

{

rhs.real++;

rhs.imaginary++;

return rhs;

}

Complex operator++(Complex &rhs, int)

{

Complex temp(rhs);

rhs.real++;

rhs.imaginary++;

return temp;

}

std::ostream &operator<<(std::ostream &os, const Complex &rhs)

{

os << "(" << rhs.real << " + " << rhs.imaginary << "i"

<< ")" << std::endl;

return os;

}

int main()

{

Complex c1(1, 2), c2(3, 4), c(9, 200);

c1.Output();

c2.Output();

c1 += c2;

c1.Output();

Complex c3 = c1 + c2;

Complex c4 = c1 += c2, c5, c6;

c3.Output();

c5 = ++c4;

c4.Output();

c5.Output();

c6 = c4++;

c4.Output();

c6.Output();

c2 = -c2;

std::cout << c2;

std::cout << c;

return 0;

}

# 실행 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

# 소스 코드 (Styled Format)

#include <iostream>

class Complex

{

private:

double real;

double imaginary;

public:

Complex(double \_real) : real(\_real), imaginary(0)

{

}

Complex(double \_real = 0, double \_imaginary = 0) : real(\_real), imaginary(\_imaginary)

{

}

void Output()

{

std::cout << real << " + " << imaginary << "i" << std::endl;

}

Complex &operator+=(const Complex &);

Complex &operator-();

friend Complex operator+(const Complex &, const Complex &);

friend Complex operator++(Complex &);

friend Complex operator++(Complex &, int);

friend std::ostream &operator<<(std::ostream &os, const Complex &);

};

Complex &Complex::operator+=(const Complex &rhs)

{

real += rhs.real;

imaginary += rhs.imaginary;

return \*this;

}

Complex &Complex::operator-()

{

real = -real;

imaginary = -imaginary;

return \*this;

}

Complex operator+(const Complex &lhs, const Complex &rhs)

{

Complex temp(lhs);

temp += rhs;

return temp;

}

Complex operator++(Complex &rhs)

{

rhs.real++;

rhs.imaginary++;

return rhs;

}

Complex operator++(Complex &rhs, int)

{

Complex temp(rhs);

rhs.real++;

rhs.imaginary++;

return temp;

}

std::ostream &operator<<(std::ostream &os, const Complex &rhs)

{

os << "(" << rhs.real << " + " << rhs.imaginary << "i"

<< ")" << std::endl;

return os;

}

int main()

{

Complex c1(1, 2), c2(3, 4), c(9, 200);

c1.Output();

c2.Output();

c1 += c2;

c1.Output();

Complex c3 = c1 + c2;

Complex c4 = c1 += c2, c5, c6;

c3.Output();

c5 = ++c4;

c4.Output();

c5.Output();

c6 = c4++;

c4.Output();

c6.Output();

c2 = -c2;

std::cout << c2;

std::cout << c;

return 0;

}