A-1

Report

제목 : C++프로그래밍

과제 8번

과제내용

p431 연습문제 7, 8번

p433-434 프로그래밍 과제 – 1~4번

미수행과제내역

담당교수 : 배성현 교수님

학과명 : 생명과학과

학번 : 5355002

이름 : 정유진

**일반 연습문제 과제**

431p

7. 시간(어떤 시각에서 어떤 시각까지의 사이)를 나타내는 클래스 Time을 작성하여 보자. Time은 hours를 멤버로 가진다.

class Time{

int hours;

public:

Time(){ hours = 0; }

Time(int h){ hours = h; }

};

① + 연산자를 프렌드 함수 형태로 중복 정의하여 보자.

② + 연산자를 멤버 함수 형태로 중복 정의하여 보자.

③ 출력 연상자인 << 을 중복정의해보라.

④Time 클래스에 분을 나타내는 멤버 변수인 minutes를 추가하여서 다시 작성하여 보라.

1) 소스코드

#include <iostream>

using namespace std;

class Time {

int hours; // 시간

int minute; // 분

public:

Time() { hours = 0; minute = 0; }

Time(int h, int m) { hours = h; minute = m; }

friend Time operator+(const Time& t1, const Time& t2);

Time operator+(const Time& t2);

void operator+=(const Time& t2);

friend ostream& operator<< (ostream& os, const Time& t);

};

Time operator+(const Time&t1, const Time& t2) { // +연산자 프렌드함수 중복

Time t;

t.hours = t1.hours + t2.hours;

t.minute = t1.minute + t2.minute;

if (t.minute >= 60) {

int rem = t.minute / 60;

t.hours += rem;

t.minute %= 60;

}

return t;

}

Time Time::operator+(const Time& t2) { // +연산자 멤버함수 중복

Time t;

t.hours = this->hours + t2.hours;

t.minute = this->minute + t2.minute;

if (t.minute >= 60) {

int rem = t.minute / 60;

t.hours += rem;

t.minute %= 60;

}

return t;

}

void Time::operator+=(const Time& t2) { // +=연산자 멤버함수 중복

this->hours = this->hours + t2.hours;

this->minute = this->minute + t2.minute;

if (this->minute >= 60) {

int rem = this->minute / 60;

this->hours += rem;

this->minute %= 60;

}

}

ostream& operator<<(ostream& os, const Time& t) { // <<연산자 프렌드함수 중복

os << t.hours << "시 " << t.minute << "분" << endl;

return os;

}

int main() {

Time t1(1, 20);

Time t2(2, 50);

Time t3;

t1 += t2;

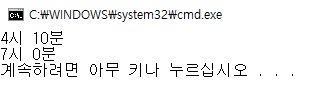
t3 = t2 + t1;

cout << t1 << t3;

return 0;

}

2) 출력결과



3) 소스코드 설명

시간 클래스에서 +, << 연산자를 중복선언하고 +연산자를 프렌드 함수 형태로 중복정의하여 결과를 출력하는 프로그램입니다. m의 값이 60 이상일 경우에 h에 m을 60으로 나눈 값을 더하고 m을 다시 60으로 나누어 남는 값을 m에 반환해줍니다. <<연산자를 이용하여 객체를 전달해 결과를 출력하였습니다. +연산자를 중복정의했기에 +=연산자 또한 중복정의하였습니다.

8. 복소수를 나타내는 Complex 클래스를 가지고 실습하여 보자.

class Complex {

private:

double real, imag;

public:

Complex(double r, double i) : real{r}, imag{i} {}

Complex operator+(Complex);

Complex operator-(Complex);

};

① 연산자 +와 -에 대한 함수를 클래스 외부에 중복 정의하라.

② 복소수 (1.0, 7.0)와 (3.0, 8.0)을 나타내는 객체를 생성하고 두개의 복소수를 + 연산자를 이용하여 합하는 문장을 완성하라.

③ a, b가 Complex 객체라고 했을 때 a + b 대신에 a.operator+(b)의 형태로도 호출이 가능한지를 확인하라.

④ +=, -= 연산자도 중복 정의해보라. 객체를 생성하여서 테스트하라.

⑤ 다음과 같은 문장을 가능하게 하려면 어떤 생성자가 정의되어야 하는가? 필요한 생성자를 정의하여 보자.

⑥ (double) 형변환 연산자를 중복 정의하여 보자. (double)이 복소수에 적용되면 복소수의 절대값을 구해서 반환한다.

⑦ 연산자 +와 -에 대한 연산자 중복 함수를 프렌즈 함수의 형태로 정의하여 보자. 사용 방법을 보여라.

⑧ 연산자 >>와 <<도 중복 정의하여 보자.

⑨ 연산자 ==과 !=도 중복 정의하여 보자.

1) 소스코드

#include <iostream>

#include <math.h>

using namespace std;

class Complex {

private:

double real, imag;

public:

Complex() { real = 0.0, imag = 0.0; }

Complex(double r, double i) : real{r}, imag{i} {}

Complex operator+(const Complex&); // 문제1

Complex operator-(const Complex&);

void operator+=(const Complex&); // 문제4

void operator-=(const Complex&);

operator double(); // 문제6

friend ostream& operator<<(ostream& os, const Complex& c); // 문제8

friend istream& operator>>(istream& os, Complex& c);

bool operator==(const Complex&); // 문제9

bool operator!=(const Complex&);

};

Complex Complex::operator+(const Complex& c2) { // +연산자 중복정의

Complex c;

c.real = this->real + c2.real;

c.imag = this->imag + c2.imag;

return c;

}

Complex Complex::operator-(const Complex& c2) { // -연산자 중복정의

Complex c;

c.real = this->real - c2.real;

c.imag = this->imag - c2.imag;

return c;

}

void Complex::operator+=(const Complex& c2) { // +=연산자 중복정의

this->real = this->real + c2.real;

this->imag = this->imag + c2.imag;

}

void Complex::operator-=(const Complex& c2) { // -=연산자 중복정의

this->real = this->real - c2.real;

this->imag = this->imag - c2.imag;

}

Complex::operator double() { // 형변환 연산자 중복정의

double n;

n = (this->real \* this->real) + (this->imag\*this->imag);

return sqrt(n);

}

ostream& operator<<(ostream& os, const Complex& c) { // <<연산자 중복정의

if (c.imag >= 0) // 허수부분이 0이상일때 출력

os << c.real << "+" << c.imag << "i" << endl;

else

os << c.real << "-" << "(" <<c.imag << "i)" << endl;

return os;

}

istream& operator>>(istream& os, Complex& c) { // >>연산자 중복정의

return os;

}

bool Complex::operator==(const Complex& c2) { // ==연산자 중복정의

return (this->real == c2.real) && (this->imag == c2.imag);

}

bool Complex::operator!=(const Complex& c2) { // !=연산자 중복정의

return (this->real != c2.real) && (this->imag != c2.imag);

}

int main() {

cout.setf(cout.boolalpha);

Complex c1(1.0, 7.0);

Complex c2(3.0, 8.0);

Complex c3, c4;

c3 = c1 + c2; // 문제2

c4 = c1.operator+(c2); // 문제3

cout << "c3 = c1 + c2 : " << c3;

cout << "c4 = c1.operator+(c2) : " << c4;

c3 += c3; // 문제5

cout << "(c3+=c3) : " << c3;

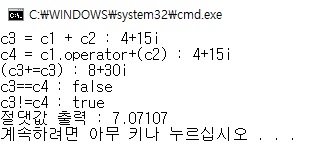
cout << "c3==c4 : " <<(c3 == c4) << endl;

cout << "c3!=c4 : " << (c3 != c4) << endl;

cout << "절댓값 출력 : " << (double)c1 << endl;

}

2) 출력결과



3) 소스코드 설명

복소수 클래스를 통해 다양한 연산자를 중복정의하여 그 결과를 출력하는 프로그램입니다. 출력 결과에서는 확인할 수 없지만 복소수가 0미만 일시에 n+(-ni) 으로 출력될 수 있게 <<연산자 중복정의에 정의해 놓았습니다. 그리고 중복정의한 연산자들이 어떤식으로 작동하는지 출력결과를 문제별로 순서대로 출력하였습니다. 5번 문제는 이해하지 못해 풀지 못했습니다.

**프로그래밍 과제**

\*

다음의 Box 클래스를 사용하여 1번부터 4번까지의 문제를 해결하라.

class Box {

private:

double length;

double width;

double height;

public:

Box(double l = 0, double w=0, double h=0) : length{l}, width{w}, height{h}{}

double getVolume(void) {

return length \* width \* height;

}

};

433p

1. Box 객체에 대하여 다음과 같이 +연산을 정의하고자 한다. + 연산을 하면 상자의 길이, 폭, 높이가 서로 합쳐진다고 가정하자. + 연산자 중복 함수를 Box 클래스의 멤버 함수로 작성해보자.

Box a(10, 10, 10), b(20, 20, 20), c;

c = a + b;

2. Box 객체에 대하여 다음과 같이 == 연산을 정의하고자 한다. == 연산을 하면 2개의 상자가 동일한 부피인지를 검사한다. – 연산자 중복 함수를 Box 클래스의 멤버 함수로 작성해 보자.

Box a(10, 10, 10), b(20, 20, 20), c;

c = a == b;

3. Box 객체에 대하여 다음과 같이 < 연산을 정의하고자 한다. < 연산에서 첫 번째 상자의 부피가 두 번째 상자보다 작으면 참을 반환한다. 그렇지 않으면 거짓을 반환한다. < 연산자 중복 함수를 Box 클래스의 멤버 함수로 작성해 보자.

Box a(10, 10, 10), b(20, 20, 20), c;

c << ( a < b );

4. Box 객체에 대하여 다음과 같이 프렌즈 함수 printBox()를 정의해 보자 .printBox()는 박스 객체의 모든 데이터를 화면에 출력한다.

void printBox( Box Box ) {

…

}

1) 소스코드

#include <iostream>

using namespace std;

class Box {

private:

double length; // 길이

double width; // 넓이

double height; // 높이

public:

Box(double l = 0, double w=0, double h=0) : length{l}, width{w}, height{h}{}

double getVolume(void) {

return length \* width \* height;

}

double getLength() { return length; }

double getWidth() { return width; }

double getHeight() { return height; }

Box operator+(const Box& b2); // 1번문제

bool operator==(const Box& b2); // 2번문제

bool operator<(const Box& b2); // 3번문제

friend void printBox(Box b); // 4번문제

};

Box Box::operator+(const Box& b2) { // +연산자 중복정의

Box b;

b.length = this->length + b2.length;

b.width = this->width + b2.width;

b.height = this->height + b2.height;

return b;

}

bool Box::operator==(const Box& b2) { // == 연산자 중복정의

return (this->length == b2.length) && (this->width == b2.width) && (this->height == b2.height);

}

bool Box::operator<(const Box& b2) { // < 연산자 중복정의

return (this->length < b2.length) && (this->width < b2.width) && (this->height < b2.height);

}

void printBox(Box b) { // friend 함수를 이용하여 박스 정보 출력

cout << "상자의 길이 : " << b.length << endl;

cout << "상자의 너비 : " << b.width << endl;

cout << "상자의 높이 : " << b.height << endl;

cout << "상자의 부피 : " << b.getVolume() << endl;

}

int main() {

Box a(10, 10, 10);

Box b(20, 20, 20);

Box c;

c = a + b;

cout << "상자 #1" << endl;

printBox(a);

cout << "상자 #2" << endl;

printBox(b);

cout << "상자 #3" << endl;

printBox(c);

/\*

cout.setf(cout.boolalpha); cout << (a < b) << endl; // <연산자 중복정의 확인

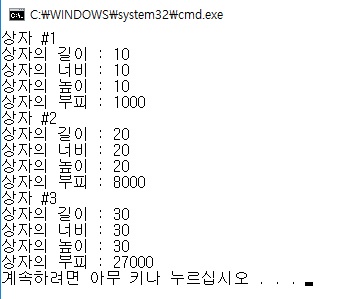
cout.setf(cout.boolalpha); cout << (a == b) << endl; // ==연산자 중복정의 확인

\*/

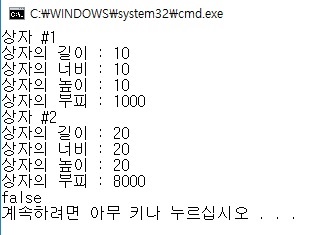
return 0;

}

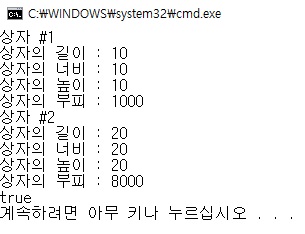
2.1) 출력결과



2.2) 출력결과



2.3) 출력결과



3) 소스코드 설명

Box 클래스를 활용하여 푸는 1, 2, 3, 4번 문제를 하나의 소스코드에 통합하여 결과를 출력해주는 프로그램입니다. 출력결과는 중간중간 주석처리를 이용하여 문제에 맞는 결과를 출력하였습니다.

4) 의견 및 고찰

문제 수는 적었지만 한 문제 안에 제시하는 것들이 많아 많은 시간을 소요했습니다. 8번 문제의 5번 조건은 Complex a = 6; 에서 a.real = 3, a.imag = 0이라는 의미가 되는지 이해하지 못하여 풀지 못했습니다. 그 외의 문제들은 큰 어려움을 겪지 않았습니다. 다만 bool이 리턴해주는 값을 true나 false로 표시하는 방법을 잊어 Cplusplus 홈페이지를 검색하여 코드를 참조하였습니다.