[ 13주차 실습과제 ]



21.11.25

1826074 오현진

1번

문제 : 연산자 중복을 프렌즈 함수로 구현하기

코드 :

#include <iostream>

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

**class** Power {

**int** kick;

**int** punch;

**public**:

Power(**int** kick=0, **int** punch=0) {

**this**->kick = kick; **this**->punch = punch;

}

**void** show();

**friend** Power& **operator**++(Power& op);

**friend** Power **operator**++(Power&op, **int** x);

};

**void** Power::show() {

cout << "kick=" << kick << ',' << "punch=" << punch << endl;

}

Power& **operator**++(Power& op) {

op.kick++;

op.punch++;

**return** op;

}

Power **operator**++(Power& op, **int** x) {

Power tmp = op;

op.kick++;

op.punch++;

**return** tmp;

}

**class** Power2 {

**int** kick;

**int** punch;

**public**:

Power2(**int** kick=0, **int** punch=0) {

**this**->kick = kick; **this**->punch = punch;

}

**void** show();

**friend** Power2& **operator**--(Power2& op);

**friend** Power2 **operator**--(Power2&op, **int** x);

};

**void** Power2::show() {

cout << "kick=" << kick << ',' << "punch=" << punch << endl;

}

Power2& **operator**--(Power2& op) {

op.kick--;

op.punch--;

**return** op;

}

Power2 **operator**--(Power2& op, **int** x) {

Power2 tmp = op;

op.kick--;

op.punch--;

**return** tmp;

}

**int** main() {

Power a(3,5), b;

b = ++a;

a.show();

b.show();

b = a++;

a.show();

b.show();

cout << endl;

Power2 c(3,5), d;

d = --c;

c.show();

d.show();

d = c--;

c.show();

d.show();

}

실행 화면 :

텍스트이(가) 표시된 사진

자동 생성된 설명

평가 : ++연산자와 –연산자의 중복을 프렌즈로 잘 구현하였습니다.

2번

문제 : private 상속 사례에 대한 문제 해결

코드 :

#include <iostream>

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

**class** Base {

**public**:

**int** a;

**void** setA(**int** a) { **this**->a = a; }

**void** showA() { cout << a << endl; }

};

**class** Derived : **public** Base {

**public**:

**int** b;

**void** setB(**int** b) { **this**->b = b; }

**void** showB() { cout << b << endl; }

};

**int** main() {

Derived x;

x.a = 5;

x.setA(10);

x.showA();

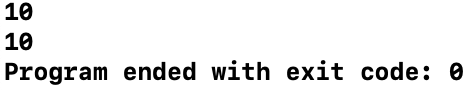
x.b = 10;

x.setB(10);

x.showB();

}

실행 화면 :



평가 : public 멤버가 아닌 모든 요소들을 public멤버로 바꾸어 접근 가능하게 바꾸어 에러를 해결했습니다.

3번

문제 : 판매실적 계산을 위한 프로그램 작성

코드 :

#include <iostream>

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

**class** Shops {

**public**:

**int** money; // 품목에 대한 가격

**int** cnt; // 판매 수량

**int** dis; // 디스카운트 율

**int** sales(); // 판매 금액

**int** saleswith(); // 판매이익(세금포함)

**int** tax(); // 세금

**int** snt(); // 실 판매이익

};

**int** Shops::sales() {

**return** money \* cnt;

}

**int** Shops::saleswith() {

**return** money \* cnt \* dis / 100;

}

**int** Shops::tax() {

**return** money \* cnt \* dis / 10;

}

**int** Shops::snt() {

**return** (money \* cnt \* dis / 100) - (money \* cnt \* dis / 100 / 10);

}

**int** main() {

**int** i = 1;

**int** sum = 0; // 합계

**int** totsaleswith = 0; // 총판매이익(세금포함)

**int** tottax = 0; // 총세금

**int** totsnt = 0; // 총 실판매이익

**int** avg; // 영업점별 평균 실판매이익

**while** (i<=3) {

Shops shop;

cout << "품목에 대한 가격 : ";

cin >> shop.money;

cout << "판매 수량 : ";

cin >> shop.cnt;

cout << "이익률 : ";

cin >> shop.dis;

cout << endl;

cout << "영업점(" << i << ")" << endl;

cout << "\*\* 판매 금액 \*\*" << endl;

cout << " - 가격 : " << shop.money << " 수량 : " << shop.cnt << " 판매 금액 : " << shop.sales() << endl;

cout << "\*\* 판매이익 \*\*" << endl;

cout << " - 이익율 : " << shop.dis << " 판매이익(세금포함) : " << shop.saleswith() << endl;

cout << "\*\* 실 판매이익 \*\*" << endl;

cout << " - 세금 : " << shop.tax() << "(10%)"

<< " 실 판매이익 : " << shop.snt() << endl;

cout << "-----------------------------------------------" << endl;

sum += shop.saleswith();

cout << "합 계 : " << sum << endl;

cout << "----------------------" << endl;

totsaleswith += shop.saleswith();

tottax += shop.tax();

totsnt += shop.snt();

i += 1;

}

avg = totsnt / 3;

cout << "(1) 총판매이익(세금포함) : " << totsaleswith << endl;

cout << "(2) 총세금 : " << tottax << endl;

cout << "(3) 총 실판매이익 : " << totsnt << endl;

cout << "(4) 영업점별 평균 실판매이익 : " << avg << endl;

}

실행 화면 :

텍스트, 영수증이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

평가: 각 매장에 대한 수치들을 계산하는 부분들을 클래스를 통해서 구현했습니다.

3번 반복하는 while문을 통하여 자동적으로 Shops클래스의 객체를 세번 생성하게 하였고, 각 반복마다 객체에 대한 가격, 수량, 이익률을 직접 입력하여 필요한 수치들에 대한 연산을 진행하도록 하였습니다.

(실행 예시에서는 똑같은 수치들을 세번 집어넣어 실행해서, 입력 및 출력이 모두 같습니다. 다르게 집어넣으면 다르게 잘 실행되지만, 마땅히 생각나는 예시가 없어 주어진 수치들을 똑같이 세번 입력했습니다.)

각 반복마다 마지막 누적치에 대한 연산도 수행하여, 실행 결과 마지막 부분에 누적치들을 출력하도록 하였습니다.