HW3 실습 (과제 아닙니다, 제출하지 않습니다.)

질문

pemta81718@gmail.com

1(a) 다음과 같은 **makefile**을 작성하라.
명령어 : cat makefile / vi makefile

```
hw3a:   hw3a.o polya.o
        g++ -o hw3a hw3a.o polya.o
        hw3a.o polya.o:polya.h
hw3b:   hw3b.o polyb.o
        g++ -o hw3b hw3b.o polyb.o
        hw3b.o polyb.o:polyb.h
```

(b) 2개의 다항식을 가진 다음과 같은 입력 파일 poly.in을 작성하라.
명령어: cat poly.in / vi poly.in

(첫 줄은 항의 수가 4이고 각 항이 x^8과 -7x^5과 -x^3과 -3의 4항으로 이루어진 다항식 이고 둘째 줄은 항의 수가 3이고 각 항이 x^5과 2x^3과 -4인 다항식이다.)

```
4       1.0 8     -7.0 5     -1.0 3       -3.0 0
3       1.0 5      2.0 3     -4.0 0
```

2. 다항식의 구현

(a) 다음 같이 동작하는 프로그램을 작성하려 한다.

make hw3a

hw3a < poly.in

```
[B611107@localhost hw2]$ hw2a < poly.in
x^8 -7x^5 -x^3 -3
x^5 +2x^3 -4
x^8 -6x^5 +x^3 -7
```

[주의] 위와 같이 출력 시 지수가 0 이 아닌 항의 경우 그 계수가 1 이면 출력을 생략하고, -1이면 -로 출력되어야 한다.

(b) main 프로그램(hw3a.cpp)은 다음과 같다.

명령어: vi hw3a.cpp

```
#include <iostream>
using namespace std;
#include "polya.h"
int main() {
    Polynomial p1, p2;

    cin >> p1 >> p2; // 2개의 다항식을 읽어 들인다.
    Polynomial p3 = p1 + p2;
    cout << p1 << p2 << p3;
}
```

(c) 다음 프로그램을 polya.h로 저장하라.
명령어: vi polya.h

```
#ifndef POLYNOMIAL_H
#define POLYNOMIAL_H
using namespace std;
class Polynomial; // 전방참조
class Term{
        friend class Polynomial;
        friend ostream& operator<<(ostream&, Polynomial&);
        friend istream& operator>>(istream&, Polynomial&);
        private:
        float coef;   // coefficient
        int exp;     // exponent
};
class Polynomial {
public:
        Polynomial();   // construct a polynomial p(x) = 0.
        Polynomial operator+(Polynomial&); // 다항식의 합을 반환
        void NewTerm(const float, const int);
        friend ostream& operator<<(ostream&, Polynomial&);
        friend istream& operator>>(istream&, Polynomial&);
private:
        Term *termArray;
        int capacity; // 1로 초기화
        int terms;      // 저장된 항의 수로 0으로 초기화
};
#endif
```

(d) 다음 프로그램 polya.cpp를 완성하라.
명령어: vi polya.cpp

```
#include <iostream>
#include "polya.h"
using namespace std;

istream& operator>> (istream& is, Polynomial& p){
// #terms and (coefficoent, exponent)의 pair들을 읽어들인다.
// 높은차수의 항부터 저장되었다고 가정한다.
        int noofterms; float coef; int exp;
        is >> noofterms;
        for (int i = 0; i < noofterms; i++){
                is >> coef >> exp; // 계수와 지수 pair를 읽어들인다.
                p.NewTerm(coef, exp);
        }
        return is;
}
ostream& operator<< (ostream& os, Polynomial& p)
{
        // 출력부분
        return os;
}
Polynomial::Polynomial() :capacity(1), terms(0){
        termArray = new Term[capacity];
}
void Polynomial::NewTerm(const float theCoeff, const int theExp)
{
        // 다항식 뒤에 새로운 항을 추가하는 함수
}
Polynomial Polynomial::operator+(Polynomial& b)
{
        //다항식의 덧셈을 해주는 함수

}
```

3. 이제 다항식의 곱을 수행하는 함수를 구현하여 동작 시켜보자
먼저 poly2.in을 준비하자
명령어: cat poly2.in / vi poly2.in

3 -1.0 4 3.0 1 -6.0 0
3 2.0 2 1.0 1 2.0 0

(a) 다음 같이 다항식 곱하기를 수행하는 프로그램을 작성하려 한다
make hw3b
hw3b < poly2.in

```
[B611107@localhost hw2]$ hw2b < poly2.in
-x^4 +3x -6
2x^2 +x +2
-2x^6 -x^5 -2x^4 +6x^3 -9x^2 -12
```

[주의] 위와 같이 출력 시 지수가 0 이 아닌 항의 경우 그 계수가 1 이면 출력을 생략하고, -1이면 -로 출력되어야 한다.

(b) main 프로그램(hw3b.cpp)은 다음과 같다.
명령어: vi hw3b.cpp

```
#include <iostream>
using namespace std;
#include "polyb.h"                // include file을 바꾸었음
int main() {
    Polynomial p1, p2;

    cin >> p1 >> p2; // 2개의 다항식을 읽어 들인다.
    Polynomial p3 = p1 * p2;    // +를 *로 바꾸었음
    cout << p1 << p2 << p3;
}
```

(c) polyb.h를 작성한다. polyb.h는 polya.h와 동일하나 Polynomial 클래스내에서 다음과 같은 public함수 선언을 추가하면 된다.
명령어: vi polyb.h

```
    Polynomial operator*(Polynomial&);
```

(d) polyb.cpp를 구현하라. polya.cpp와 거의 같으나 include file을 polya.h대신 polyb.h로 하고, 다음 함수를 추가로 구현하여야 한다. 다항식의 합을 이용하도록 하면 편리하다.
명령어: vi polyb.cpp

```
Polynomial Polynomial::operator*(Polynomial& b)
{


}
```