1. 구조체 배열로 다항식의 표현 만들기

계수와 지수를 담을 수 있는 구조체를 만들고, 구조체 배열을 만들어 다항식의 표현을 나타낼 수 있도록 하였다.

```
typedef struct TERM {
        double coef;
        int expon;
}TERM;
void append(TERM* poly, double coef, int expon, int arr);
void sort(TERM* poly, int n);
void print(TERM* poly, int n);
```

main 함수에서 배열을 생성하고 append 함수를 통해 배열과 계수, 지수 값, 배열의 인덱스 값을 전달해 해당 인덱스 값을 갖는 배열에 초기화 시켜주었다. Sort 함수는 다항식을 지수가 큰 것에서 작은 것으로 정렬한다. print 함수는 다항식의 표현을 출력한다.

2. 링크드리스트로 다항식의 표현 만들기

링크드 리스트로 각 항을 노드로 만들고 연결하여 다항식의 표현을 할 수 있도록 구성하였다.

```
typedef struct NODE {
        double coef;
        int expon;
        struct NODE* link;
}NODE;

typedef struct NODELINK { //노드들을 연결해주기 위한 구조체
        struct NODE* head;
        struct NODE* tail;
    }NODELINK;
```

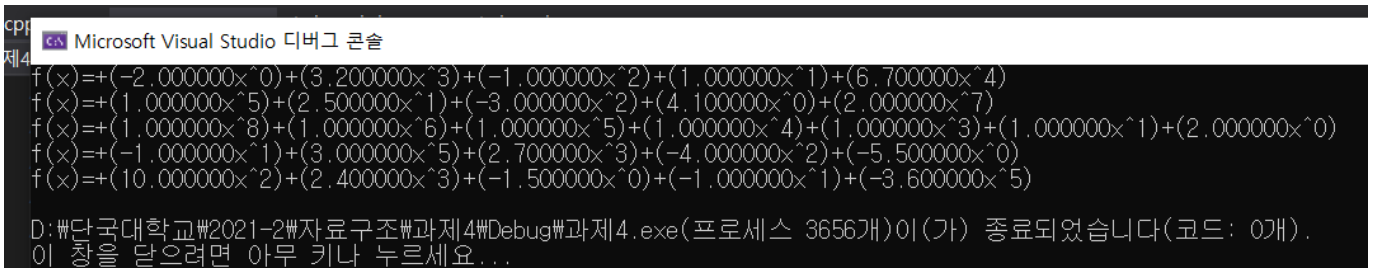
위와 같이 계수와 지수를 갖고, 다음 노드를 가르킬 NODE 포인터를 갖는 NODE 구조체를 만들고 연결된 노드의 가장 앞과 뒤를 가르키는 NODELINK 구조체를 만들었다.

```
void append(NODELINK* poly, double coef, int expon);
void erase(NODELINK* poly);
void sort(NODELINK* poly);
void print(NODELINK* poly);
```

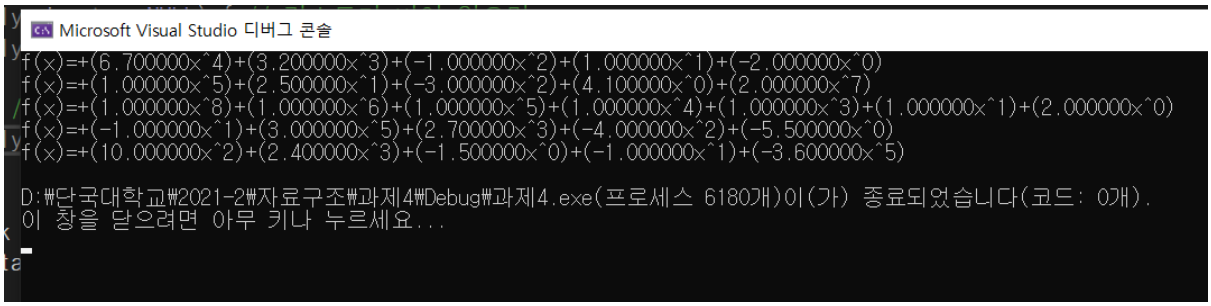
그리고 다음과 같은 함수를 만들어 다항식의 표현을 생성하는데 사용했다. 먼저 append 함수로 노드를 동적으로 할당 받아 추가하고 이전 노드와 연결시킨다. erase 함수는 동적으로 할당 받은 노드를 해제시키는 기능을 하고, sort 함수는 다항식을 지수가 큰 것에서 작은 것으로 정렬한다. print 함수는 다항식의 표현을 출력한다.

3. 5개이상의 항을 갖는 다항식 5개로 테스트 해보기

먼저 구조체 배열을 사용해 다항식의 표현을 나타내는 것을 테스트해본 결과이다.



다음은 링크드 리스트를 사용해 다항식의 표현을 나타내는 것을 테스트해본 결과이다.



결과는 똑같이 나왔다. 하지만 다항식을 만들 때 미리 정해진 배열의 크기만을 사용할 수 있고, 초기화를 위해 배열의 인덱스까지 매개변수로 넘겨줘야 하는 구조체 배열보다는 동적으로 노드를 생성해 사용하여 언제든 노드를 추가할 수 있는 링크드리스트가 더 편리하다고 생각했다. 또한

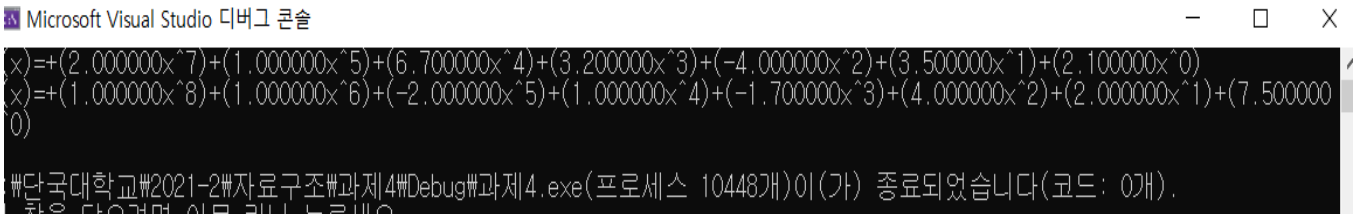
4. 다항식의 가감하는 함수 추가하여 테스트 해보기

다항식의 가감을 하는 함수를 만들어 테스트 해보았다.

다항식의 덧셈은 f(x)= 6.7x^4 + 3.2x ^ 3 – x ^ 2 + x – 2 와 f(x) = x^5 + 2.5x -3x^2 + 4.1 + 2x^7 다항식을 사용해 서로 더하는 것을 테스트 해보았고, 다항식의 뺄셈은 f(x) = x^8 + x^6 + x^5 + x^4 + x^3 + x + 2 와 f(x) = -x + 3x^5 + 2.7x^3 - 4x^2 -5.5 다항식을 사용해 전자에서 후자를 빼는 것으로 테스트 해보았다.

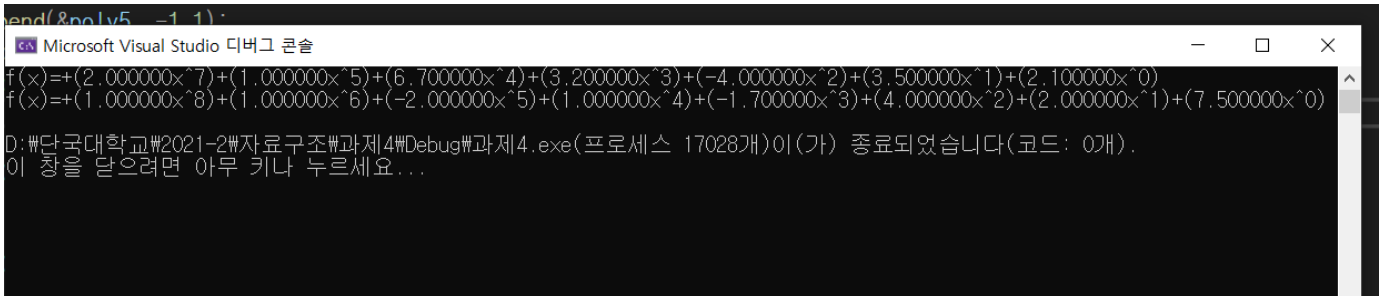
먼저 구조체 배열을 사용해 테스트 해본 결과이다.

```
void poly_add(TERM* a, int asize, TERM* b, int bsize, TERM* c);
void poly_minu(TERM* a, int asize, TERM* b, int bsize, TERM* c);
```



다음은 링크드리스트 자료구조를 사용해 다항식의 표현한 경우 테스트 결과이다.

```
NODELINK* poly_add(NODELINK* a, NODELINK* b);
NODELINK* poly_minu(NODELINK* a, NODELINK* b);
```



결과는 같게 나왔으나 다항식의 가감을 수행하는 함수를 동작시킬 때 구조체 배열을 사용하면 가감 후 다항식의 항이 몇 개인지 미리 알고 있어야 배열의 크기를 초기화 하거나 동적 할당할 수 있었다. 따라서 유동적으로 노드를 추가하고 연결하는데 유용한 자료구조는 링크드 리스트라는 것을 깨 달았다.

5. 다항식의 승제도 가능할지 토의해보기

다항식의 곱셈과 나눗셈의 경우 모든 항들을 서로 연산해주기 때문에 항이 같은 지 비교할 필요가 없어 sort를 굳이 할 필요도 없고 더 쉽게 수행할 수 있을 것이라고 생각한다. 하지만 항이 많아 지기 때문에 위에 말했듯이 수행 완료된 항의 개수를 알고 있어야 하는 구조체 배열은 더욱 사용하기 어려워질 것이라고 생각하여 연결리스트를 이용하는 것이 더 편하고 효율적일 것이다.