

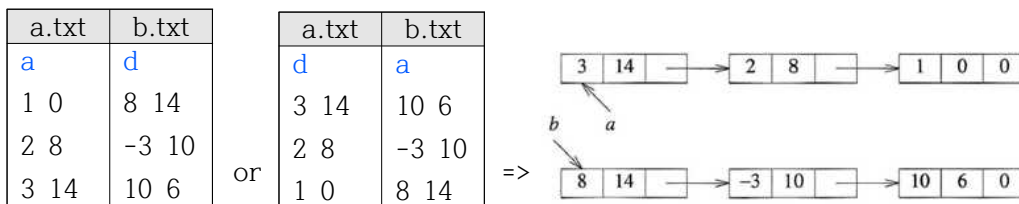
# 자료구조응용

## 10. 연결 리스트 : 다항식

1. 다음과 같이 chain을 이용한 다항식 더하기 프로그램을 작성하라.

(1) 실행 순서

① 두 개의 입력파일("a.txt," "b.txt")로부터 데이터를 입력받아서 두 개의 다항식을 chain 형태로 구현한다. 아래 예는  $a = 3x^{14} + 2x^8 + 1$ ,  $b = 8x^{14} - 3x^{10} + 10x^6$  에 대한 입력 예이다.



※ 첫 줄 입력이 'a'이면 지수(exponent) 차수에 대해 오름차순(ascending order), 'd'이면 내림차순(descending order)으로 입력됨. 오름차순으로 입력되면 각 노드는 chain의 첫 노드로 삽입되어야 하며, 내림차순으로 입력되면 각 노드는 chain의 마지막 노드로 추가됨

- ② a, b 두 다항식의 정보를 출력한다.
- ③ a+b의 결과를 c에 저장하는 다항식 더하기를 실행한다.
- ④ 다항식 c를 출력한다.
- ⑤ 다항식 a, b, c를 모두 삭제한다.

(2) 구현 세부사항

```
typedef struct polyNode *polyPointer;
typedef struct polyNode {
    int coef;
    int expon;
    polyPointer link;
} polyNode;
polyPointer a,b;
```

coef	expon	link
------	-------	------

### (3) 실행 예



```
선택 Microsoft Visual Studio 디버그 콘솔
a :
(000002726A24BAC0 : +3x^14 : 000002726A24BA20)
(000002726A24BA20 : +2x^8 : 000002726A24B890)
(000002726A24B890 : +1x^0 : 0000000000000000)

b :
(000002726A2578D0 : +8x^14 : 000002726A257A10)
(000002726A257A10 : -3x^10 : 000002726A257560)
(000002726A257560 : +10x^6 : 0000000000000000)

a+b=c :
(000002726A2580F0 : +11x^14 : 000002726A257A60)
(000002726A257A60 : -3x^10 : 000002726A257970)
(000002726A257970 : +2x^8 : 000002726A258050)
(000002726A258050 : +10x^6 : 000002726A257B50)
(000002726A257B50 : +1x^0 : 0000000000000000)

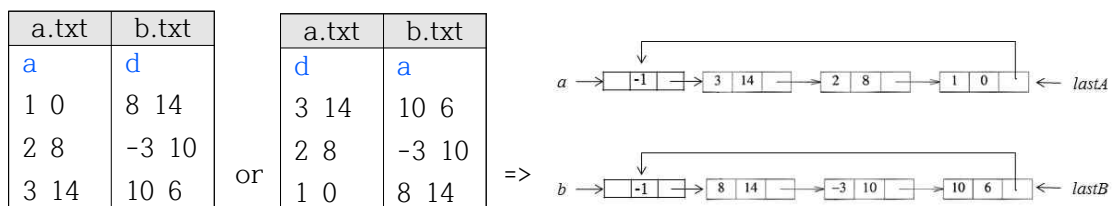
G:\₩2020년₩2020년1학기₩자료구조응용₩DSA-10₩₩x64₩Debug₩1.exe(23444 프로세스)이(가) 0 코드로 인해 종료되었습니다.
이 창을 닫으려면 아무 키나 누르세요.
```

2. 다음과 같이 헤더노드를 가진 단일 환형연결리스트 (singly linked circular list)을 이용한 다항식 더하기 프로그램을 작성하라.

### (1) 실행 순서

① 입력파일("a.txt," "b.txt")로부터 데이터를 입력받아서 두 개의 다항식 a, b를 각각 헤더노드를 가진 단일 환형연결리스트 형태로 구현하고, 두 다항식을 더하시오. 더한 결과도 역시 헤더노드를 가진 단일 환형연결리스트 형태로 구현하시오. 그리고 빈 노드를 관리하는 avail 리스트에 리스트 a, b, c 순으로 반환한 후 avail 리스트를 운영체제에 반환하시오.

※  $a = 3x^{14} + 2x^8 + 1$ ,  $b = 8x^{14} - 3x^{10} + 10x^6$  에 대한 입력 예



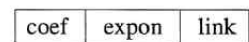
※ 첫 줄 입력이 'a'이면 지수 차수에 대해 오름차순(ascending order), 'd'이면 내림차순(descending order)으로 입력됨. 오름차순으로 입력되면 각 노드는 환형리스트의 첫 노드로 삽입되어야 하며, 내림차순으로 입력되면 각 노드는 환형리스트의 마지막 노드로 추가

- ② a, b 두 다항식의 정보를 출력한다.
- ③ a+b의 결과를 c에 저장하는 다항식 더하기를 수행한다.
- ④ 다항식 c를 출력한다.
- ⑤ 다항식 a, b, c를 **avail**에 반납한다.
- ⑥ avail 리스트를 출력한다.
- ⑦ avail을 삭제한다.

### (2) 구현 세부사항

※ 주의 : a, b, c는 **헤더노드를 가진 단일 환형연결리스트**이며, **avail**은 **단일연결리스트**임

```
typedef struct polyNode *polyPointer;
typedef struct polyNode {
    int coef;
    int expon;
    polyPointer link;
} polyNode;
polyPointer a,b;
polyPointer c, lastA, lastB, avail = NULL;
```



### (3) 실행 예

```
선택 Microsoft Visual Studio 디버그 콘솔
a :
(00000168FE3AA630 : -842150451x^-1 : 00000168FE3B7750)
(00000168FE3B7750 : +3x^14 : 00000168FE3B7BB0)
(00000168FE3B7BB0 : +2x^8 : 00000168FE3AA680)
(00000168FE3AA680 : +1x^0 : 00000168FE3AA630)

b :
(00000168FE3B77A0 : -842150451x^-1 : 00000168FE3B7A70)
(00000168FE3B7A70 : +8x^14 : 00000168FE3B7660)
(00000168FE3B7660 : -3x^10 : 00000168FE3B7340)
(00000168FE3B7340 : +10x^6 : 00000168FE3B77A0)

a+b=c :
(00000168FE3B7570 : -842150451x^-1 : 00000168FE3B7B60)
(00000168FE3B7B60 : +11x^14 : 00000168FE3B78E0)
(00000168FE3B78E0 : -3x^10 : 00000168FE3B7390)
(00000168FE3B7390 : +2x^8 : 00000168FE3B7D40)
(00000168FE3B7D40 : +10x^6 : 00000168FE3B7930)
(00000168FE3B7930 : +1x^0 : 00000168FE3B7570)

avail :
(00000168FE3B7B60 : +11x^14 : 00000168FE3B78E0)
(00000168FE3B78E0 : -3x^10 : 00000168FE3B7390)
(00000168FE3B7390 : +2x^8 : 00000168FE3B7D40)
(00000168FE3B7D40 : +10x^6 : 00000168FE3B7930)
(00000168FE3B7930 : +1x^0 : 00000168FE3B7570)
(00000168FE3B7570 : -842150451x^-1 : 00000168FE3B7A70)
(00000168FE3B7A70 : +8x^14 : 00000168FE3B7660)
(00000168FE3B7660 : -3x^10 : 00000168FE3B7340)
(00000168FE3B7340 : +10x^6 : 00000168FE3B77A0)
(00000168FE3B77A0 : -842150451x^-1 : 00000168FE3B7750)
(00000168FE3B7750 : +3x^14 : 00000168FE3B7BB0)
(00000168FE3B7BB0 : +2x^8 : 00000168FE3AA680)
(00000168FE3AA680 : +1x^0 : 00000168FE3AA630)
(00000168FE3AA630 : -842150451x^-1 : 0000000000000000)

G:\₩2020년₩2020년1학기₩자료구조응용₩DSA-10₩x64₩Debug₩2.exe(19748 프로세스)이(
가) -1073741819 코드로 인해 종료되었습니다.
이 창을 닫으려면 아무 키나 누르세요.
```

#### ■ 제출 형식

- 솔루션 이름 : DS\_10
- 프로젝트 이름 : 1, 2
- 솔루션 폴더를 압축하여 제출할 것.
- 학습관리시스템에 과제를 올릴 때 제목:  
1차 제출: 학번\_이름\_DS\_10(1), 2차 제출: 학번\_이름\_DS\_10(2)  
제출은 2회걸쳐 가능(수정 시간 기준으로 처리)