

자료구조응용

09. 연결 리스트 : 다항식

1. 다음과 같이 정렬되지 않는 점수(정수 데이터)를 입력에 대하여 Circularly Linked List를 만들고 실행 예와 같이 수행되는 프로그램을 작성하라.

(1) 실행 순서

① 입력파일("input.txt")로 부터 데이터를 입력받으면서 Circularly Linked List를 만든다.

input.txt
50 80 30 20 19 90
30 55 77 30 87 7

② Circularly Linked List의 처음부터 끝까지 노드의 데이터를 출력한다.

출력 형태 : (노드주소, 데이터 필드, 링크값)

③ 성적이 홀수인 노드를 Circularly Linked List에서 삭제한다.

④ Linked List의 처음부터 끝까지 노드의 데이터를 출력한다.

출력 형태 : (노드주소, 데이터 필드, 링크값)

⑤ Linked List를 모두 삭제한다.

(2) 구현 세부사항

① 구조체 정의문은 다음과 같다.

```
typedef struct listNode *listPointer;  
typedef struct listNode {  
    int data;  
    listPointer link;  
} listNode;  
listPointer first = NULL;
```

실행 예

```
C:\WINDOWS\system32\cmd.exe  
The Circularly Linked List contains:  
(00F164B0, 50, 00F164E8 )(00F164E8, 80, 00F16558 )(00F16558, 30, 00F16130 )  
(00F16130, 20, 00F16590 )(00F16590, 19, 00F16168 )(00F16168, 90, 00F162B8 )  
(00F162B8, 30, 00F162F0 )(00F162F0, 55, 00F16360 )(00F16360, 77, 00F16830 )  
(00F16830, 30, 00F16868 )(00F16868, 87, 00F168A0 )(00F168A0, 7, 00F164B0 )  
  
After deleting nodes with odd value  
The Circularly Linked List contains:  
(00F164B0, 50, 00F164E8 )(00F164E8, 80, 00F16558 )(00F16558, 30, 00F16130 )  
(00F16130, 20, 00F16168 )(00F16168, 90, 00F162B8 )(00F162B8, 30, 00F16830 )  
(00F16830, 30, 00F164B0 )  
계속하려면 아무 키나 누르십시오 . . .
```

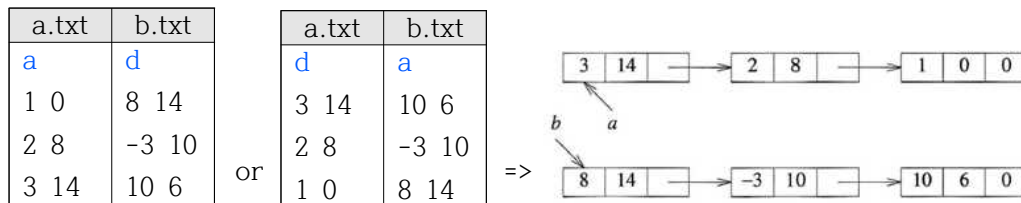
2. 다음과 같이 chain을 이용한 다항식 더하기 프로그램을 작성하라.

(1) 함수정의

- ① findLast : 체인의 마지막 노드를 찾는 함수
- ② Program 4.2 insert : temp->data=50, if블록(non-empty list에 추가)수정
- 맨 앞에 삽입할 경우 고려
- ③ inputPoly : 파일로부터 다항식 생성하기. findLast, insert 호출
- ④ Program 4.4 printList : 참고
- ⑤ Program 4.9 padd : 참고
- ⑥ Program 4.10 attach : 참고
- ⑦ Program 4.11 erase : 참고

(2) 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
- ※ a 인 경우, 항상 마지막 노드로 삽입하여 구현한 chain에 대해 invert 함수를 수행해도 됨

- 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
------	-------	------

```

void insert(listPointer *first, listPointer x)
{
    /* insert a new node with data = 50 into the chain
       first after node x */
    listPointer temp;
    MALLOC(temp, sizeof(*temp));
    temp->data = 50;
    if (*first) {
        temp->link = x->link;
        x->link = temp;
    }
    else {
        temp->link = NULL;
        *first = temp;
    }
}

```

Program 4.2: Simple insert into list



```

// modified insert
void insert(...)
{
    // creation of a node
    ...
    if(...)
    { // add to non-empty list
        if(...)
        { // as a first node
        }
        else
        { // as an intermediate or a last node
        }
    }
    else
    { // add to empty list
    }
}

```

```

void printList(listPointer first)
{
    printf("The list contains: ");
    for (; first; first = first->link)
        printf("%4d", first->data);
    printf("\n");
}

```

Program 4.4: Printing a list

```

polyPointer padd(polyPointer a, polyPointer b)
/* return a polynomial which is the sum of a and b */
polyPointer c, rear, temp;
int sum;
MALLOC(rear, sizeof(*rear));
c = rear;
while (a && b)
    switch (COMPARE(a->expon, b->expon)) {
        case -1: /* a->expon < b->expon */
            attach(b->coef, b->expon, &rear);
            b = b->link;
            break;
        case 0: /* a->expon = b->expon */
            sum = a->coef + b->coef;
            if (sum) attach(sum, a->expon, &rear);
            a = a->link; b = b->link; break;
        case 1: /* a->expon > b->expon */
            attach(a->coef, a->expon, &rear);
            a = a->link;
    }
/* copy rest of list a and then list b */
for (; a; a = a->link) attach(a->coef, a->expon, &rear);
for (; b; b = b->link) attach(b->coef, b->expon, &rear);
rear->link = NULL;
/* delete extra initial node */
temp = c; c = c->link; free(temp);
return c;
}

```

Program 4.9: Add two polynomials

```

void attach(float coefficient, int exponent,
            polyPointer *ptr)
/* create a new node with coef = coefficient and expon =
   exponent, attach it to the node pointed to by ptr.
   ptr is updated to point to this new node */
polyPointer temp;
MALLOC(temp, sizeof(*temp));
temp->coef = coefficient;
temp->expon = exponent;
(*ptr)->link = temp;
*ptr = temp;
}

```

Program 4.10: Attach a node to the end of a list

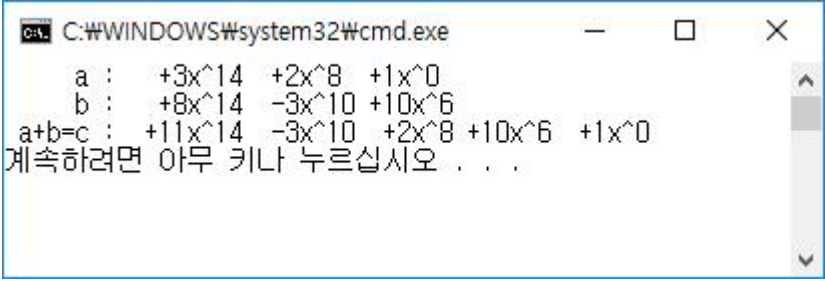
```

void erase(polyPointer *ptr)
/* erase the polynomial pointed to by ptr */
polyPointer temp;
while (*ptr) {
    temp = *ptr;
    *ptr = (*ptr)->link;
    free(temp);
}
}

```

Program 4.11: Erasing a polynomial

(3) 실행 예



```
C:\WINDOWS\system32\cmd.exe
a : +3x^14 +2x^8 +1x^0
b : +8x^14 -3x^10 +10x^6
a+b=c : +11x^14 -3x^10 +2x^8 +10x^6 +1x^0
계속하려면 아무 키나 누르십시오 . . .
```

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

(1) 함수정의

<교재 함수 수정하여 구현>

(2) 실행 순서

① 입력파일(“a.txt,” “b.txt”)로부터 데이터를 입력받아서 두 개의 다항식 a, b를 각각 헤더노드를 가진 단일 환형연결리스트 형태로 구현하고 last 포인터를 유지한다. ※ inputPolyCL 2회 호출

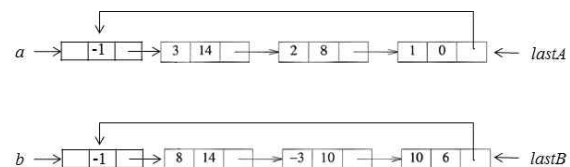
※ $a = 3x^{14} + 2x^8 + 1$, $b = 8x^{14} - 3x^{10} + 10x^6$ 에 대한 입력 예

a.txt	b.txt
a	d
1 0	8 14
2 8	-3 10
3 14	10 6

or

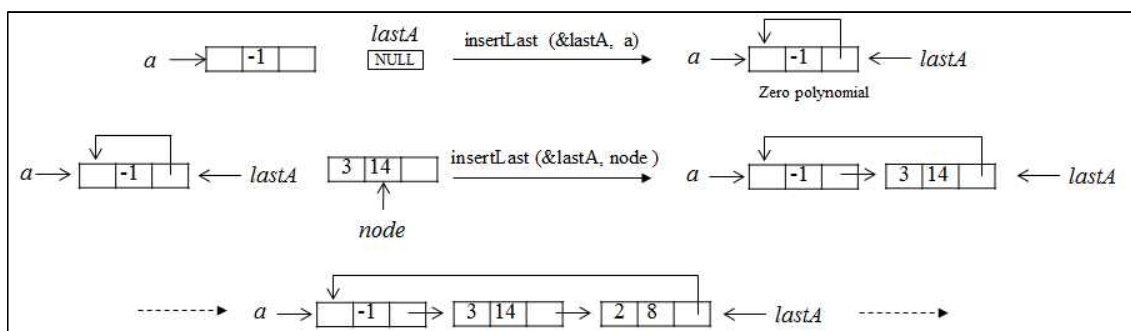
a.txt	b.txt
d	a
3 14	10 6
2 8	-3 10
1 0	8 14

=>

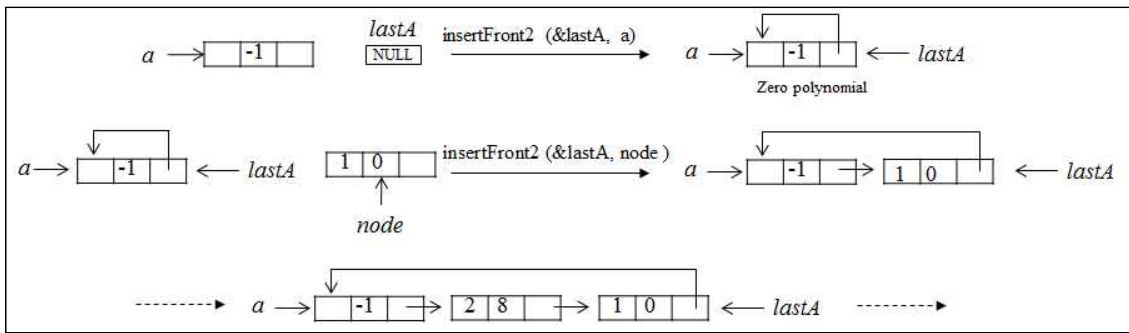


※ 첫 줄 입력이 ‘a’이면 지수 차수에 대해 오름차순(ascending order), ‘d’이면 내림차순(descending order)으로 입력됨. 오름차순으로 입력되면 각 노드는 환형리스트의 첫 노드로 삽입되어야 하며, 내림차순으로 입력되면 각 노드는 환형리스트의 마지막 노드로 추가

<내림차순 입력에 대한 다항식 생성($a = 3x^{14} + 2x^8 + 1$)>



<오름차순 입력에 대한 다항식 생성($a = 3x^{14} + 2x^8 + 1$)>



※ 주의: 위 그림의 경우라면 첫 노드(1, 0) 삽입 시 last를 변경하고 이후는 변경 없음

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

(3) 구현 세부사항

※ 주의 : a, b, c는 헤더노드를 가진 단일 환형연결리스트이며, avail은 단일연결리스트임

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

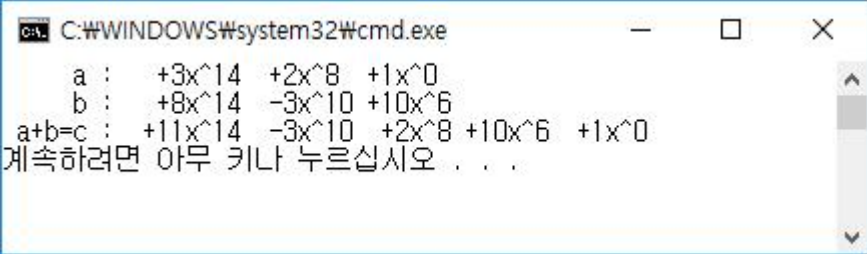
coef	expon	link
------	-------	------

```
polyPointer cpadd(polyPointer a, polyPointer b)
{
    /* polynomials a and b are singly linked circular lists
       with a header node. Return a polynomial which is
       the sum of a and b */
    polyPointer startA, c, lastC;
    int sum, done = FALSE;
    startA = a; /* record start of a */
    a = a->link; /* skip header node for a and b */
    b = b->link;
    c = getNode(); /* get a header node for sum */
    c->expon = -1; lastC = c;
    do {
        switch (COMPARE(a->expon, b->expon)) {
            case -1: /* a->expon < b->expon */
                attach(b->coef, b->expon, &lastC);
                b = b->link;
                break;
            case 0: /* a->expon == b->expon */
                if (startA == a) done = TRUE;
                else {
                    sum = a->coef + b->coef;
                    if (sum) attach(sum, a->expon, &lastC);
                    a = a->link; b = b->link;
                }
                break;
            case 1: /* a->expon > b->expon */
                attach(a->coef, a->expon, &lastC);
                a = a->link;
        }
    } while (!done);
    lastC->link = c;
    return c;
}
```

Program 4.15: Adding two polynomials represented as circular lists with header nodes

※ 기타 함수는 교재 및 강의자료 참고

(3) 실행 예



```
C:\WINDOWS\system32\cmd.exe
a : +3x^14 +2x^8 +1x^0
b : +8x^14 -3x^10 +10x^6
a+b=c : +11x^14 -3x^10 +2x^8 +10x^6 +1x^0
계속하려면 아무 키나 누르십시오 . . .
```


■ 제출 형식

- 공학인증 시스템(ABEEK)에 과제를 올릴 때 제목:
 - 1차 제출: 학번_이름_DS_09(1), 2차 제출: 학번_이름_DS_09(2)
- 솔루션 이름 : DS_09
- 프로젝트 이름 : 1, 2, 3
- 실행화면을 캡처하여 한글파일에 추가 후 솔루션 폴더에 포함.
- 한글 파일명 : 학번_이름.hwp
- 솔루션 폴더를 압축하여 제출할 것.
- 솔루션 압축 파일 명:
 - 1차 제출: 학번_이름_DS_09(1).zip, 2차 제출: 학번_이름_DS_09(2).zip
- 제출은 2회 걸쳐 가능(수정 시간 기준으로 처리)