

자료구조응용

11. 연결리스트: 다항식 (20점)

2020.4.22.(수)

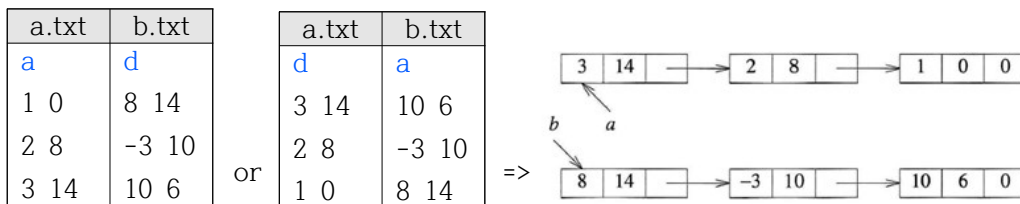
1. 다음과 같이 chain을 이용한 다항식 더하기 프로그램을 작성하라. (6점)

(1) 함수정의

- ① findLast : 노드의 마지막을 찾는 함수
- ② insert : 자료구조응용 10에서 구현한 insert 함수 사용
- ③ inputPoly : 파일로부터 다항식 생성하기. [findLast], insert 호출
- ④ Program 4.4 printList : 일부 수정
- ⑤ Program 4.9 padd : 수정 없음
- ⑥ Program 4.10 attach : 수정 없음
- ⑦ Program 4.11 erase : 수정 없음

(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
- ※ 주의: 입력파일의 마지막에 엔터키를 추가하라!

- 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, int data)
{ /* 체인의 x 노드 바로 뒤에 data 값을 가지는 새 노드를 추가 */

    /* 새 노드 할당 및 데이터 저장 */
    listPointer temp;
    MALLOC(temp, sizeof(*temp));
    temp->data = data;

    if(*first == NULL)
    { /* (a) 빈 리스트에 노드 추가 */

        _____
        _____
    }
    else
    { /* 비어 있지 않은 리스트에 추가 */
        if ( x == NULL )
        { // (b) 첫 노드로 추가

            _____
            _____
        }
        else
        { // (c) 두 번째 이상 위치에 추가

            _____
            _____
        }
    }
}

```

```

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

```

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

```

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 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 . . .

```

2. 다음과 같이 헤더노드를 가진 단일 환형연결리스트 (singly linked circular list)을 이용한 다항식 더하기 프로그램을 작성하라. (8점)

(1) 함수정의

<교재 함수 수정하여 구현>

① insertFront2

Program 4.18 insertFront를 수정. 입력되는 순서대로 리스트의 처음, 즉 헤더노드 바로 다음에 추가된다. 노드 추가 시 getNode() 호출

② insertLast

Program 4.18 insertFront를 수정. 입력되는 순서대로 리스트의 마지막에 추가된다. insertFront 함수의 else 블록 마지막에 `*last = node;` 추가함으로써 간단히 구현됨. 노드 추가 시 getNode() 호출

③ inputPolyCL

파일로부터 “헤더노드를 가진 단일 환형연결리스트”로 된 다항식을 생성함

④ printCList

Program 4.19를 수정. 헤더노드를 제외한 항들만 출력되도록 할 것.

⑤ attach

Program 4.10를 수정. `MALLOC(temp, sizeof(*temp));` 대신에 `temp = getNode();`를 사용

<교재 함수 그대로 구현 : Program 4.11 ~ 4. 15>

① erase, ② getNode, ③ retNode, ④ cerase, ⑤ cpadd

(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

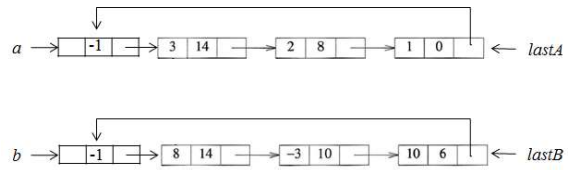
※ $a = 3x^{14} + 2x^8 + 1$, $b = 8x^{14} - 3x^{10} + 10x^6$ 에 대한 입력 예

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

or

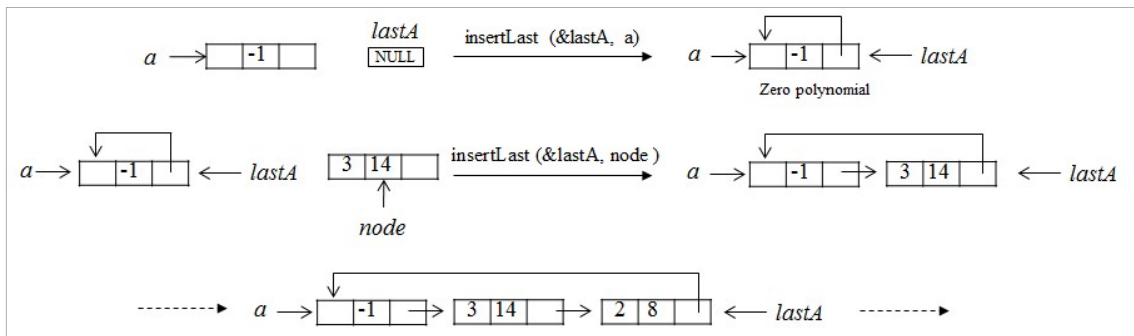
a.txt	b.txt
d	a
3 14	10 6
2 8	-3 10
1 0	8 14

=>

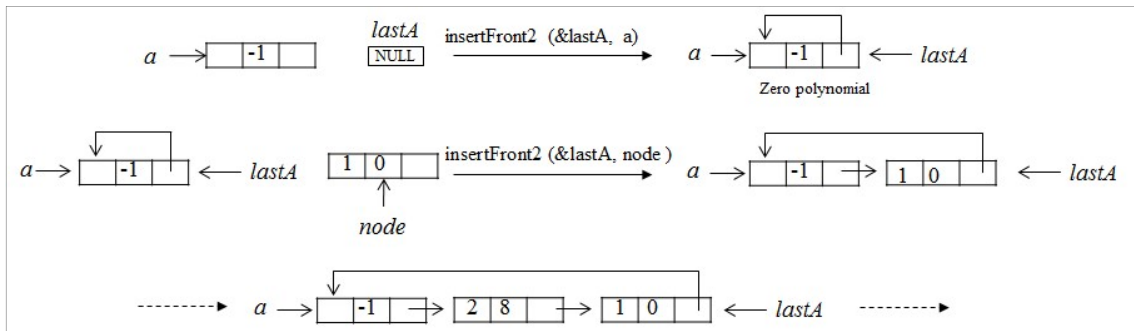


※ 첫 줄 입력이 'a'이면 지수 차수에 대해 오름차순(ascending order), 'd'이면 내림차순(descending order)으로 입력됨. 오름차순으로 입력되면 각 노드는 환형리스트의 첫 노드로 삽입되어야 하며, 내림차순으로 입력되면 각 노드는 환형리스트의 마지막 노드로 추가

<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

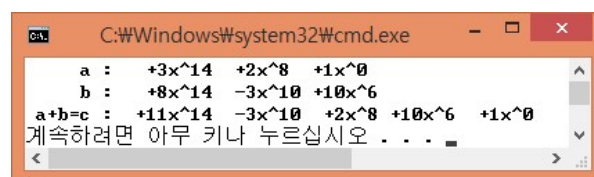
```
typedef struct polyNode *polyPointer;
typedef struct polyNode {
    int coef;
    int expon;
    polyPointer link;
} polyNode;
polyPointer a, b;
polyPointer c, lastA, lastB, avail = NULL;
```

```
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 . . .
```

3. 다음과 같이 정수 데이터를 입력하면서 “헤더노드를 가진 이중연결환형리스트 (doubly linked circular list)”를 만들고 실행예와 같이 수행되는 프로그램을 작성하라. (6점)

(1) 실행 순서

① 입력파일(“input.txt”)로 부터 데이터를 입력받는 순서대로 이중환형연결리스트의 마지막 노드로 추가 되도록 한다.

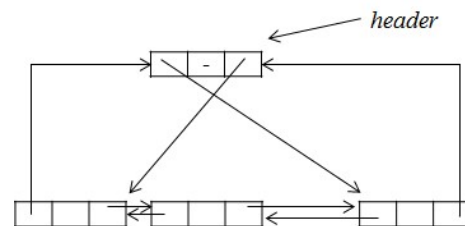
50	80	30	20	19	90	30	55	77	30
99	45	55	89	91	10	20	66	38	59
22	55	88	22	66	29	50	95	78	83

- ② 순방향과 역방향으로 노드의 데이터를 출력한다. (forward & backward)
- ③ 성적이 50점 이하인 노드를 삭제한다.
- ④ 순방향과 역방향으로 노드의 데이터를 출력한다.
- ⑤ 헤더를 제외한 모든 노드를 삭제한다.

(2) 구현 세부사항

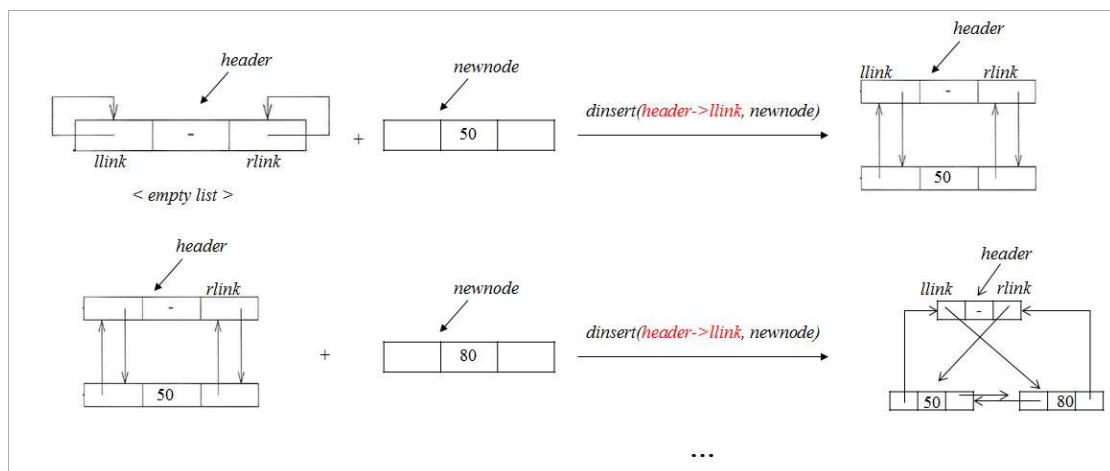
① 구조체 선언문

```
typedef struct node *nodePointer;
typedef struct node {
    nodePointer llink;
    int data;
    nodePointer rlink;
} node;
nodePointer header = NULL;
```



② 이중연결환형리스트의 Last node로 추가하기

- empty list를 생성한 후 새로운 노드를 하나씩 추가해 간다.
- header node의 llink는 last node pointer 역할을 한다. (* dinsert 함수 호출)



③ 함수정의

```
void dinsert(nodePointer node, nodePointer newnode)
{/* insert newnode to the right of node */
    newnode->llink = node;
    newnode->rlink = node->rlink;
    node->rlink->llink = newnode;
    node->rlink = newnode;
}
```

Program 4.26: Insertion into a doubly linked circular list

```
void ddelete(nodePointer node, nodePointer deleted)
{/* delete from the doubly linked list */
    if (node == deleted)
        printf("Deletion of header node not permitted.\n");
    else {
        deleted->llink->rlink = deleted->rlink;
        deleted->rlink->llink = deleted->llink;
        free(deleted);
    }
}
```

Program 4.27: Deletion from a doubly linked circular list

- 기타 함수들을 적절하게 선언하고 사용할 것(자료구조생성, 리스트 출력, 삭제관련함수 등)

(3) 디버깅 및 실행 예

이름	값	형식
header	0x012e7138 _header	node *
llink	0x00263420 {link=0x00264cd0 data=-842150451 rlink=0x00264cd0}	node *
data	-842150451	int
rlink	0x00263468 {link=0x00263420 data=50 rlink=0x002644f0}	node *
llink	0x00263420 {link=0x00264cd0 data=-842150451 rlink=0x00264cd0}	node *
data	50	int
rlink	0x002644f0 {link=0x00263468 data=80 rlink=0x00264538}	node *
llink	0x00263468 {link=0x00263420 data=50 rlink=0x002644f0}	node *
data	80	int
rlink	0x00264538 {link=0x002644f0 data=30 rlink=0x00264580}	node *
llink	0x002644f0 {link=0x00263468 data=80 rlink=0x00264538}	node *
data	30	int
rlink	0x00264580 {link=0x00264538 data=20 rlink=0x002645c8}	node *
llink	0x00264538 {link=0x002644f0 data=30 rlink=0x00264580}	node *
data	20	int
rlink	0x002645c8 {link=0x00264580 data=19 rlink=0x00264610}	node *
llink	0x00264580 {link=0x00264538 data=20 rlink=0x002645c8}	node *
data	19	int
rlink	0x00264610 {link=0x002645c8 data=90 rlink=0x00264658}	node *
llink	0x002645c8 {link=0x00264580 data=19 rlink=0x00264610}	node *
data	90	int
rlink	0x00264658 {link=0x00264610 data=30 rlink=0x002646a0}	node *
llink	0x00264610 {link=0x002645c8 data=90 rlink=0x00264658}	node *
data	30	int
rlink	0x002646a0 {link=0x00264658 data=55 rlink=0x002646e8}	node *


```
C:\Windows\system32\cmd.exe

After creating a doubly linked circular list with a head node :
forward
50 80 30 20 19 90 30 55 77 30
99 45 55 89 91 10 20 66 38 59
22 55 88 22 66 29 50 95 78 83

backward
83 78 95 50 29 66 22 88 55 22
59 38 66 20 10 91 89 55 45 99
30 77 55 30 90 19 20 30 80 50

After deleting numbers less than and equal to 50 :
forward
80 90 55 77 99 55 89 91 66 59
55 88 66 95 78 83

backward
83 78 95 66 88 55 59 66 91 89
55 99 77 55 90 80

After deleting all nodes except for the header node :
forward

backward

계속하려면 아무 키나 누르십시오 . . .
```

■ 제출 형식

- 솔루션 이름 : DS 11
- 프로젝트 이름 : 1, 2, 3
- 각 소스파일에 주석처리
“학번 이름”
“본인은 이 소스파일을 복사 없이 직접 작성하였습니다.”
- 솔루션 정리 후 솔루션 폴더를 “학번.zip”으로 압축하여 과제 게시판에 제출

■ 주의

- 소스 복사로는 실력향상을 기대할 수 없습니다!!!
- 마감 : 4월 26일(일) 자정