

자료구조응용

11. 연결리스트: 다항식 덧셈 (10점)

2022.4.11.

1. 다음과 같이 체인을 이용한 다항식 더하기 프로그램을 작성하라. (10점)

[실행순서]

- ① 두 개의 다항식 데이터 파일("a.txt," "b.txt")로부터 입력받아서 각 다항식을 체인 형태로 구현한다.
 - 각 파일의 첫 줄이 'a'이면 지수에 대해 오름차순, 'd'이면 내림차순으로 입력된다.
 - 다항식을 표현하는 체인은 내림차순이어야 하므로, 오름차순으로 입력되는 각 항은 체인의 첫 노드로 삽입되어야 하며, 내림차순으로 입력되는 각 항은 체인의 마지막 노드로 추가되어야 한다. 'a' 입력인 경우, 입력되는 각 항을 체인의 마지막 노드로 삽입하여 구현한 후 invert 함수를 수행해도 된다.

$a = 3x^{14} + 2x^8 + 1$, $b = 8x^{14} - 3x^{10} + 10x^6$ 에 대한 입력 예					
a.txt	b.txt		a.txt	b.txt	
a	d		d	a	
1 0	8 14		3 14	10 6	
2 8	-3 10		2 8	-3 10	
3 14	10 6	or	1 0	8 14	
$a \rightarrow [3 \mid 14] \rightarrow [2 \mid 8] \rightarrow [1 \mid 1 \mid 0]$					
$b \rightarrow [8 \mid 14] \rightarrow [-3 \mid 10] \rightarrow [10 \mid 6 \mid 0]$					

- 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.

[구현세부사항]

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

[함수원형]

※ 다음은 참고사항임! 각자 더 효율적인 방법이 있다면 그대로 구현하면 됨!

① void inputPoly(char *filename, polyPointer *p);

- 파일로부터 입력형식 'd' 혹은 'a'에 따라 적절하게 하나의 다항식을 생성한다. 입력형식이 'd'이면 (a) findLast 함수 호출 후 insert 함수를 호출하여 노드를 추가하거나 (b) 항상 마지막 노드에 대한 포인터를 유지하면서 attach 함수를 사용 할 수도 있다, 입력형식이 'a'이면 insert를 호출하여 항상 첫 노드로 추가하도록 한다.

② void findLast(polyPointer first, polyPointer *last);

- first가 가리키는 체인의 마지막 노드를 찾는다.

※ findLast 함수 대신 마지막 노드에 대한 포인터를 유지해도 됨.

③ void insert(polyPointer *first, polyPointer x, int coef, int expon);

- 다항식 체인 first에 대해 x 노드 바로 뒤에 (coef, expon) 다항식 항을 새 노드로 추가

※ 자료구조응용 10. 1번의 insert 함수를 활용.

④ void printList(polyPointer first);

- 체인 first를 순회하며 다항식 정보를 출력한다.

⑤ polyPointer padd(polyPointer a, polyPointer b);

- 다항식 a, b를 더한 결과 다항식을 생성하여 반환한다.

⑥ void attach(float coefficient, int exponent, polyPointer *ptr);

- (coefficient, exponent) 값을 가지는 다항식 노드를 새로 생성한 후, *ptr이 가리키는 마지막 노드 뒤에 추가하기. 노드 추가 후 *ptr은 새 노드로 수정된다.

⑦ void erase(polyPointer *ptr);

- ptr이 가리키는 체인을 삭제하는 함수이다. 체인의 처음부터 따라가며 모든 노드에 대한 메모리해제를 수행한다.

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

=>수정

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

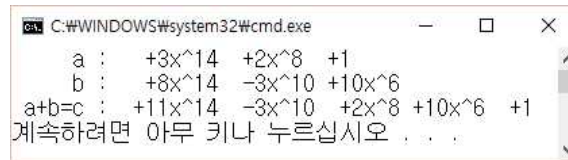
```

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

[실행예]



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

■ 제출 형식

- 솔루션 이름 : DS 11
- 프로젝트 이름 : 1
- 각 소스파일에 주석처리
“학번 이름”

“본인은 이 소스파일을 다른 사람의 소스를 복사하지 않고 직접 작성하였습니다.”

- 실행화면을 캡처한 보고서를 작성 후 pdf 파일로 변환하여 솔루션 폴더에 포함
- 솔루션 정리 메뉴를 수행 후 전체 솔루션을 “학번.zip”으로 압축하여 제출

■ 주의

- 소스 복사로는 실력향상을 기대할 수 없습니다!!!
- 1차 마감 (LMS 과제 마감일) : 수업일 자정
- 2차 마감 (LMS 과제 열람기한) : 수업 익일 자정(만점의 50%, 반올림)
- 1, 2차 마감 이외의 제출은 허용하지 않습니다. (이메일 제출 불가!)