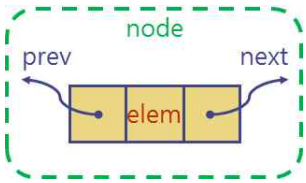
〈자료구조 실습〉 - 연결리스트

※ 입출력에 대한 안내

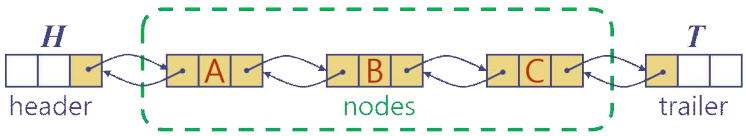
- 특별한 언급이 없으면 문제의 조건에 맞지 않는 입력은 입력되지 않는다고 가정하라.
- 특별한 언급이 없으면, 각 줄의 맨 앞과 맨 뒤에는 공백을 출력하지 않는다.
- 출력 예시에서 □는 각 줄의 맨 앞과 맨 뒤에 출력되는 공백을 의미한다.
- 입출력 예시에서 ↦ 이 후는 각 입력과 출력에 대한 설명이다.

연결리스트 참고사항 : 아래 나오는 이중연결리스트/헤더/트레일러 노드 방식을 이용해도 되고, 우리 교재에 나오는 방식을 이용해도 됩니다.

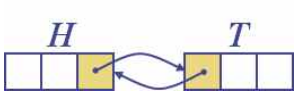
1. 연결리스트 구조
   - 각 노드에 저장되는 정보
     - elem: 원소
     - prev: 이전 노드를 가리키는 링크
     - next: 다음 노드를 가리키는 링크



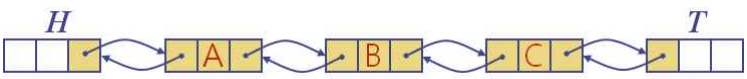
   - 헤더 및 트레일러 노드
     - 데이터를 가지지 않는 특별 노드



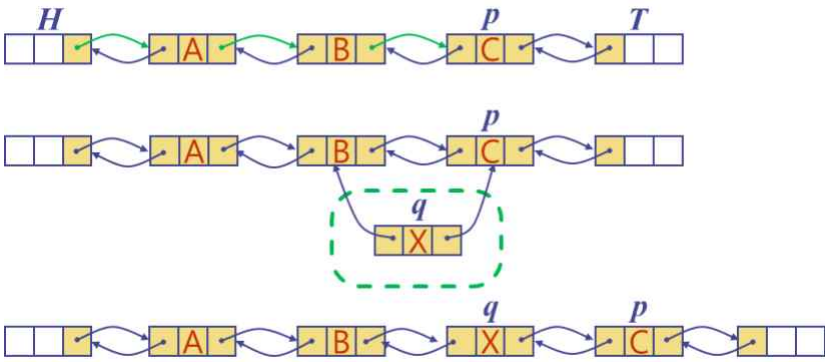
2. 이중연결리스트 초기화
   - 초기에는 헤더 및 트레일러 노드만 존재
   - **O(1)** 시간 소요



3. 이중연결리스트 순회
   - 연결리스트의 모든 원소들을 방문
   - 순회하면서 필요한 작업 수행(예를 들면 출력)
   - **O(n)** 시간 소요



4. 이중연결리스트에서 삽입
   - 이중연결리스트의 지정된 순위 **r**에 원소 **e**를 삽입
   - **O(n)** 시간 소요



5. 이중연결리스트에서 삭제
   - 이중연결리스트로부터 지정된 순위 **r**의 노드를 삭제하고, 원소를 반환
   - **O**(**n**) 시간 소요

※ 참고: 초기화, 순회, 삽입, 삭제에 관한 상세 알고리즘은 교재를 참고

[문제 1] 이중연결리스트를 이용하여 아래의 영문자 연산을 구현하시오.

- 다음 네 가지 연산을 지원해야 함 (순위는 **1**부터 시작한다고 가정)
  - **add**(**list**, **r**, **e**) : **list**의 순위 **r**에 원소 **e**를 추가한다.
  - **delete**(**list**, **r**) : **list**의 순위 **r**에 위치한 원소를 삭제한다
  - **get**(**list**, **r**) : **list**의 순위 **r**에 위치한 원소를 반환한다.
  - **print**(**list**) : **list**의 모든 원소를 저장 순위대로 공백없이 출력한다.

  ※ 순위 정보가 유효하지 않으면 화면에 에러 메시지 "invalid position" 출력하고, 해당 연산을 무시한다.

- 입력에 대한 설명 (아래 입출력 예시 참조)
  - 각 연산의 내용이 한 줄에 한 개씩 입력되고, 한 개의 줄에는 연산의 종류, 순위, 원소 순서로 입력된다.
  - **연산의 종류:** 연산 이름의 맨 앞 영문자가 대문자 **A**, **D**, **G**, **P**로 주어진다.
  - **순위:** 양의 정수
  - **원소:** 영문자(대문자, 소문자 모두 가능)

입력 예시 1

```
5          ↦ 연산의 개수: 5
A 1 S      ↦ add(list, 1, 'S')
A 2 t      ↦ add(list, 2, 't')
A 3 r      ↦ add(list, 3, 'r')
A 3 a      ↦ add(list, 3, 'a')
P          ↦ print(list)
```

출력 예시 1

```
Star       ↦ 연산 P에 의한 출력
```

입력 예시 2

```
9          ↦ 연산의 개수: 9
A 1 D      ↦ add(list, 1, 'D')
A 2 a      ↦ add(list, 2, 'a')
A 3 y      ↦ add(list, 3, 'y')
D 1        ↦ delete(list, 1)
P          ↦ print(list)
G 3        ↦ get(list, 3)
A 1 S      ↦ add(list, 1, 'S')
P          ↦ print(list)
G 3        ↦ get(list, 3)
```

출력 예시 2

```
ay
invalid position

Say
y
```

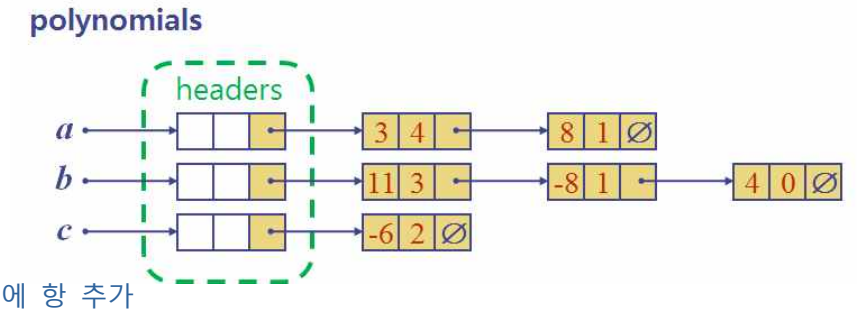
다항식 덧셈 참고사항 : 연결리스트를 이용하여 구현하되, 아래 설명 방식을 이용해도 되고, 우리 교재에 나오는 방식을 이용해도 됩니다.

1. 다항식을 표현하는 연결리스트 구조
   - 하나의 다항식(polynomial)을 하나의 **헤더 단일연결리스트**로 표현하는 방식 사용
   - 다항식의 각 항은 하나의 노드로 표현하고, 각 노드에는 다음 세 개의 필드를 저장
     - coef: 항의 계수
     - exp: 항의 차수
     - next: 다음 노드를 가리키는 링크
   - 하나의 연결리스트의 각 노드는 차수의 내림차순 순으로 유지하고, 계수가 0인 항의 노드는 유지하지 않음
   - 예) 아래 세 개의 다항식을 나타내는 단일연결리스트 그림

     a = $3x^4 + 8x$

     b = $11x^3 - 8x + 4$

     c = $-6x^2$



2. 다항식에 항 추가

◦ 기존 다항식의 마지막 항을 표현하는 노드 k에 계수 c와 차수 e로 이루어진 새 항 추가

```
Alg appendTerm(k, c, e)
   input last term of a polynomial expression k, coefficient c, exponent e
   output cxe appended to k

1. t ← getnode()
2. t.coef ← c
3. t.exp ← e
4. t.next ← NULL
5. k.next ← t
6. k ← t        {k advances to t}
7. return
```

3. 다항식 덧셈

◦ 두 개의 다항식 x, y에 대한 덧셈을 수행하여 그 결과를 새로운 헤더 단일연결리스트에 저장

- 예: 위 예의 다항식 a, b의 덧셈 결과는 $3x^4 + 11x^3 + 4$ 를 반환

```
Alg addPoly(x, y)
   input polynomial expression x, y
   output x + y
1. result  ← getnode()      {new header }
2. result.next←NULL         {may be null }
3. i ← x.next
4. j ← y.next
5. k ← result
6. while ((i ≠ NULL) & (j ≠ NULL))
      if (i.exp > j.exp)
         appendTerm(k, i.coef, i.exp)
         i ← i.next
      else if (i.exp < j.exp)
         appendTerm(k, j.coef, j.exp)
         j ← j.next
      else
         sum ← i.coef + j.coef
         if (sum ≠ 0)
            appendTerm(k, sum, i.exp)
         i ← i.next
         j ← j.next
7. while (i ≠ NULL )
      appendTerm(k, i.coef, i.exp)
      i ← i.next
8. while (j ≠ NULL)
      appendTerm(k, j.coef, j.exp)
      j ← j.next
9. return result
```

[문제 2] 다항식의 덧셈을 구하는 프로그램을 작성하라.

- 입력에 대한 설명 (아래 입출력 예시 참조)
  - 첫 번째 다항식의 항의 개수가 입력되고, 이후에 다항식의 각 항의 (계수, 지수) 쌍이 지수의 내림차순으로 입력됨
  - 동일한 방식으로 두 번째 다항식의 정보가 입력됨
- 출력에 대한 설명 (아래 입출력 예시 참조)
  - 결과 다항식의 각 항의 (계수, 지수) 쌍을 지수의 내림차순으로 출력

입력 예시 1

```
3          ↦ 첫 번째 다항식의 항의 개수
5 3 3 2 3 1    ↦ 5x^3 + 3x^2 + 3x
3          ↦ 두 번째 다항식의 항의 개수
2 6 2 3 1 0    ↦ 2x^6 + 2x^3 + 1
```

출력 예시 1

```
□2 6 7 3 3 2 3 1 1 0  ↦ 2x^6+7x^3+3x^2+3x+1
```

입력 예시 2

```
2          ↦ 첫 번째 다항식의 항의 개수
2 7 3 0      ↦ 2x^7 + 3
3          ↦ 두 번째 다항식의 항의 개수
-3 10 3 7 -3 0    ↦ -3x^10 + 3x^7 -3
```

출력 예시 2

```
□-3 10 5 7    ↦ -3x^10 + 5x^7
```