자료구조 Data Structure | 조행래

연결 리스트

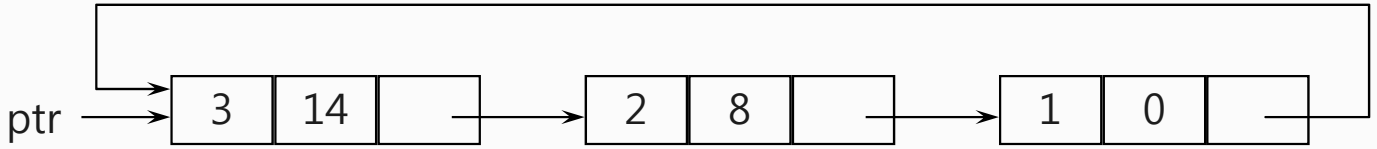
원형 연결 리스트

학습 목표

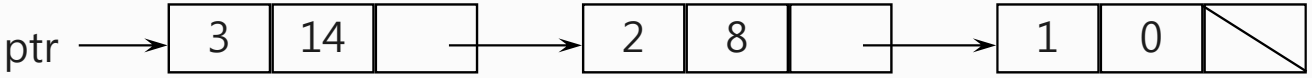
- 원형 연결 리스트의 개념을 이해한다.
- 원형 연결 리스트를 이용하여 다항식을 표현할 수 있다.

1. 원형 연결 리스트의 개념

- **원형 연결 리스트란?**
  - 마지막 노드의 link가 처음 노드를 가리키는 연결 리스트

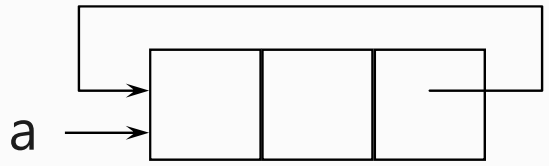


  - Chain: 마지막 노드의 link가 null인 연결 리스트

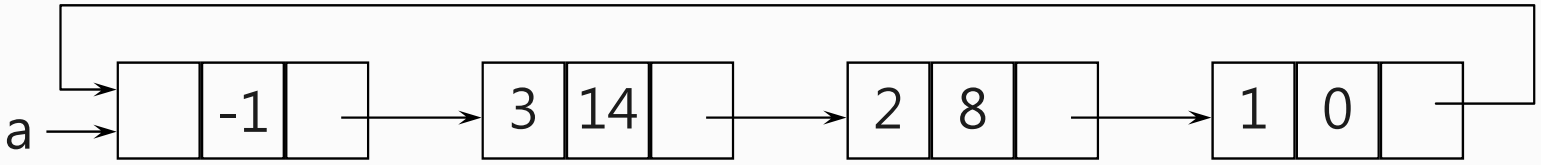


2. 헤드 노드

- 노드가 없는 원형 연결 리스트 → head node



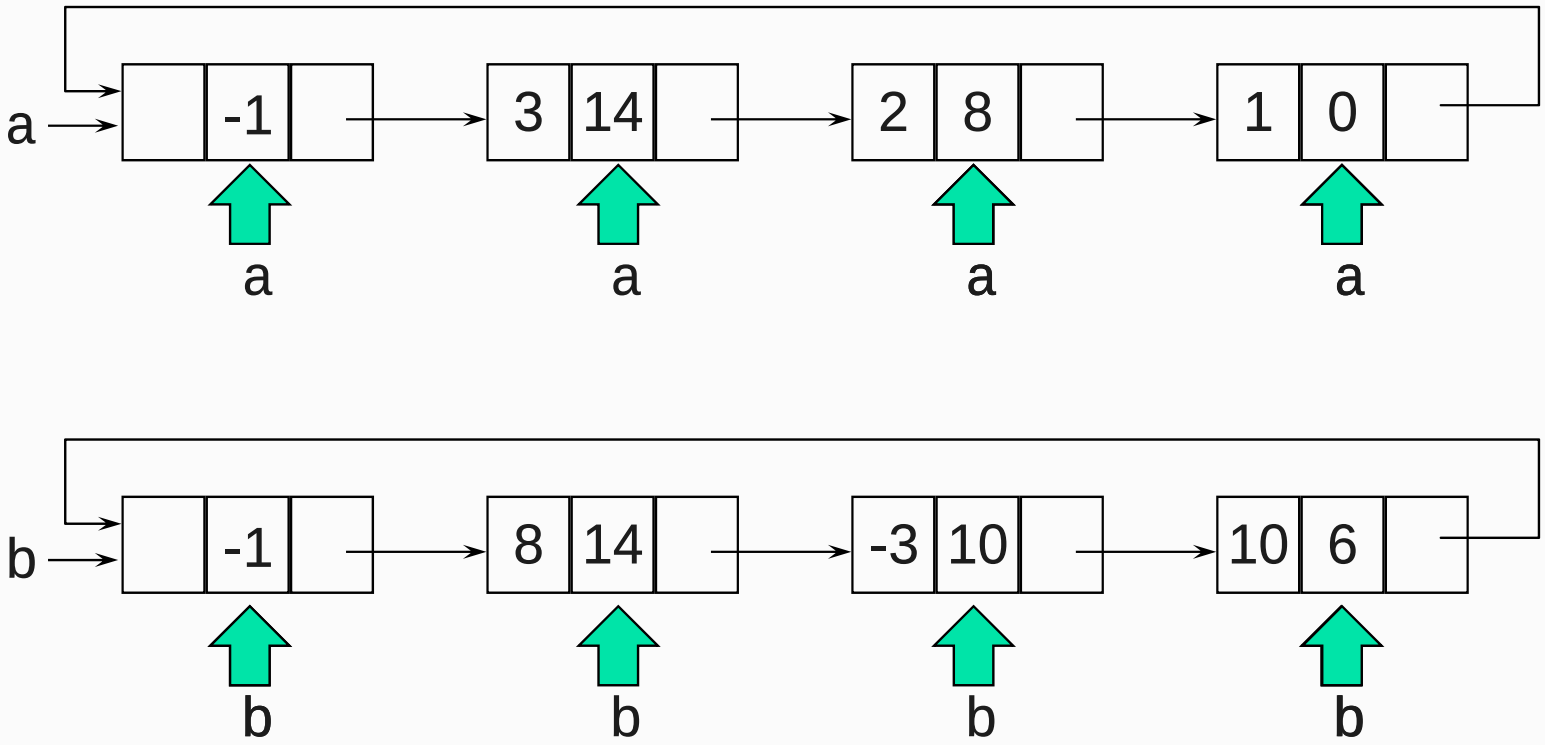
Zero polynomials



exponent가 -1

$3x^{14} + 2x^{8} + 1$

3. 다항식 더하기(원형 연결 리스트)



다항식 더하기 알고리즘

```
struct poly *cpadd(struct poly *a, struct poly *b)
{
/* 다항식 a와 b: 헤드 노드가 있는 원형 연결 리스트
    d = a + b를 계산한 후, d를 반환   */
    struct poly *starta, *d, *lastd;
    int sum, done = FALSE;
    starta = a;                                  // a의 시작 노드를 기록
    a = a→link;    b = b→link;                  // a와 b의 head node를 skip
    d = get_node();                              // 합을 위한 head node 할당
    d→expon = -1;    lastd = d;
    do   {
       switch (COMPARE(a→expon, b→expon))   {
          case -1:   // a→expon < b→expon
                     attach(b→coef, b→expon, &lastd);
                     b = b→link;  break;
```

다항식 더하기 알고리즘 (계속)

```
            case 0:    // a→expon == b→expon
                       if (starta == a) done = TRUE;
                       else   {
                                  sum = a→coef + b→coef;
                                  if (sum) attach(sum, a→expon, &lastd);
                                  a = a→link; b = b→link;
                       }
                       break;
            case: 1    // a→expon > b→expon
                       attach(a→coef, a→expon, &lastd);
                       a = a→link;
            }
    }       while ( !done );
    lastd→link  =  d;
    return  d;
}
```

요약 정리

- 원형 연결 리스트의 개념을 설명
- 원형 연결 리스트를 이용하여 다항식을 표현하고, 합을 구하는 알고리즘을 구현