

다항식 덧셈 알고리즘

- Step 0: 덧셈 결과를 기록할 빈 / 용량을 만든다.
- Step 1: 두 개의 다항식에 대해 각각 '계수'와 '지수'를 비교하여, 현재 결과 다항식에 '다항식'을 구성하는 요소로 존재하는지 여부를 판단한다.
- Step 2: 현재 다항식에 '0'이 empty를 뜻하는 Step 1을 수행한다.
- Step 3: Step 1, 2를 다항식의 끝을 알려 다항식에 add한다.

리스트 응용문제: 다항식 덧셈

- $p(x) = 4x^4 + 3x^3 + 2x^2 + 2x + 1$ 를 더하면  $p(x) + q(x) = 4x^4 + 3x^3 + 7x^2 + 2x + 4$ 가 된다.
- 이 연산을 수행하는 프로그램을 만들어 보자.
  - 다항식 p(x)를 저장하는 자료구조: [4, 4], [3, 2], [2, 0]
  - add method: 리스트 자료구조에 [coef, x] 형식의 제보 리스트를 추가한다.
  - peek method: 0번째 리스트 원소를 리턴한다.
  - pop method: 0번째 리스트 원소를 꺼내 리턴한다.
  - size method: 리스트의 길이를 리턴한다.
  - print method: 다항식 형태를 프린트 한다.

```
class Poly:
    def __init__(self):
        self.eq = []
    def add(self, coef, k):
        self.eq.append([coef, k])
    def peek(self):
        return self.eq[0]
    def pop(self):
        return self.eq.pop(0)
    def size(self):
        return len(self.eq)
    def print(self):
        for term in self.eq:
            print(term[0], "x", term[1], " + ", end=" ")
        print()

p = Poly()
p.add(4, 4)
p.add(3, 2)
p.add(3, 0)

q = Poly()
q.add(3, 3)
q.add(2, 1)
q.add(1, 0)

p.print()
q.print()

# print(p.peek())
# print(q.peek())

# print(p.pop())
# print(q.pop())

# p.print()
# q.print()
```

해당 parameter에는 '다항식' 객체가 인자값으로 들어간다.  
 두 다항식을 더한 결과의 객체도 class type이 당연히 '다항식' 이어야 한다.

→ 'a' 다항식이 남아있는 최고차항이 'b' 다항식에 남아있는 최고차항보다 클 때 ⇒ 'a'의 최고차항이 'c' 다항식에 그대로 추가됨.

← 'b' 다항식의 모든 항들이 pop되고, 'a' 다항식의 요소들만 남아 있을 때.

← 'a' 다항식의 모든 항들이 pop되고, 'b' 다항식의 요소들이 남아 있을 때.

※ 더하는 두 다항식의 최고차항 비교는 더하는 과정이 끝날때까지 진행하기 때문에, 'while' 문을 사용해야함.

① 두 다항식 전부 항들을 가지고 있을 때  
 - 최고차항이 다를 때

↳ 최단화한 값일 때.

② 두 다항식 중 하나의 다항식만 한도를 가지고 있을 때