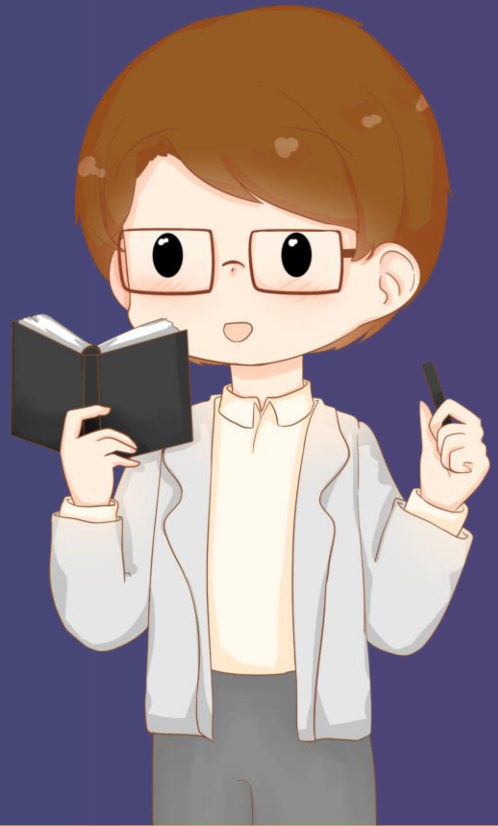
파이썬으로 배우는 알고리즘 기초

Chap 2. 분할정복

2.6

큰 정수의 계산법



2.6 큰 정수의 계산법

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

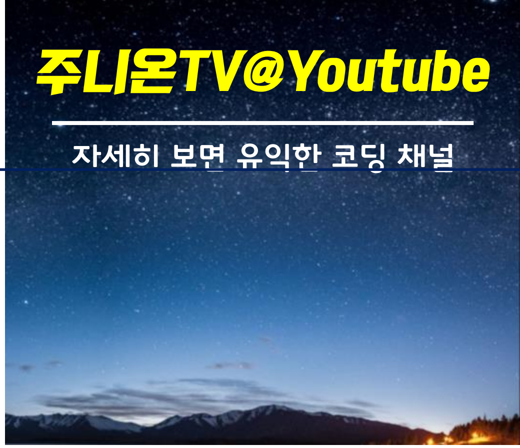
▪ 큰 정수의 산술 문제

- 문제: 특정 컴퓨터/언어가 표현할 수 없는 큰 정수의 산술 연산
- 가정: 10진수 체계에서의 덧셈과 곱셈

- 10진수를 소프트웨어적으로 표현하는 방법은?
  - 리스트를 이용하여 각 자리수(*digit*)를 하나의 원소로 저장
  - 567,832: $S = [2, 3, 8, 7, 6, 5]$

5	6	7	8	3	2
$S[5]$	$S[4]$	$S[3]$	$S[2]$	$S[1]$	$S[0]$

2.6 큰 정수의 계산법



- 큰 정수의 **덧셈**
  - n개의 자릿수(*digit*) 각각을 더하면서 올림수(*carry*)를 고려

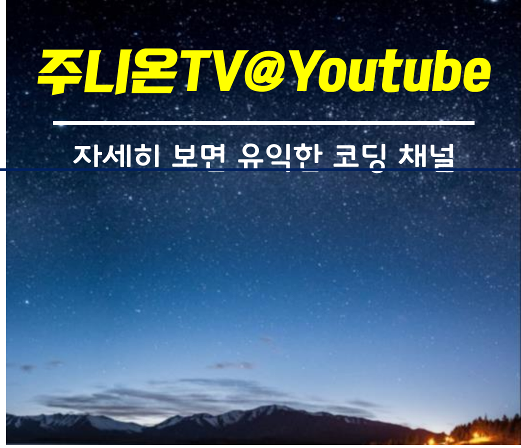
$$
\begin{array}{cccccc}
 & 1 & 1 & 0 & 0 & \leftarrow \textit{carry} \\
 & 9 & 8 & 7 & 6 & \\
+ & & 5 & 4 & 3 & \\
\hline
1 & 0 & 4 & 1 & 9 &
\end{array}
$$

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

Algorithm 2.9: Arithmetic for Large Integers

```python
def ladd (u, v):
    n = len(u) if (len(u) > len(v)) else len(v)
    result = []
    carry = 0
    for k in range(n):
        i = u[k] if (k < len(u)) else 0
        j = v[k] if (k < len(v)) else 0
        value = i + j + carry
        carry = value // 10
        result.append(value % 10)
    if (carry > 0):
        result.append(carry)
    return result
```

2.6 큰 정수의 계산법




```python
u =  [6, 7, 8, 9]
v =  [3, 4, 5]
print(9876 + 543)
print(ladd(u, v)[::-1])

u =  [2, 3, 8, 7, 6, 5]
v =  [3, 2, 7, 3, 2, 4, 9]
print(567832 + 9423723)
print(ladd(u, v)[::-1])
```

2.6 큰 정수의 계산법

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

- 큰 정수의 곱셈: 단순무식한 (Brute-Force) 방법
  - 초등학교에서 배운 방법 $\in \Theta(n^2)$

$$
\begin{array}{cccccc}
 & & & 1 & 2 & 3 \\
\times & & & & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
+ & & 4 & 9 & 2 & \\
\hline
 & & 5 & 5 & 3 & 5
\end{array}
$$

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

- **큰 정수의 곱셈: 분할정복(Divide-and-Conquer)**
  - n개의 자릿수(digit)로 된 숫자를 $n/2$개의 자릿수로 분할
  - 둘 중 하나의 자릿수는 $\lceil n/2 \rceil$ 이고, 다른 하나는 $\lfloor n/2 \rfloor$ 가 됨

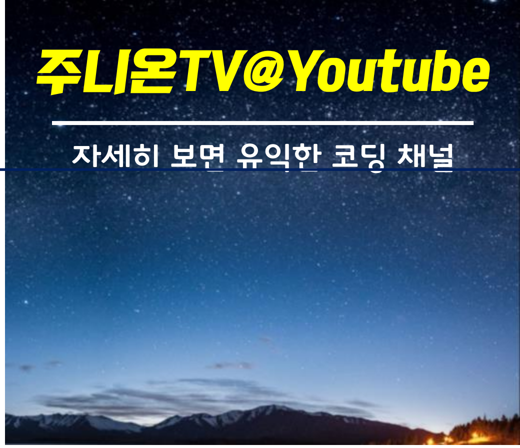
$$\underbrace{567{,}832}_{\text{6 digits}} = \underbrace{567}_{\text{3 digits}} \times 10^3 + \underbrace{832}_{\text{3 digits}}$$

$$\underbrace{9{,}423{,}723}_{\text{7 digits}} = \underbrace{9{,}423}_{\text{4 digits}} \times 10^3 + \underbrace{723}_{\text{3 digits}}$$

$$\underbrace{u}_{n \text{ digits}} = \underbrace{x}_{\lceil n/2 \rceil \text{ digits}} \times 10^m + \underbrace{y}_{\lfloor n/2 \rfloor \text{ digits}}$$

- **10의 지수:** $m = \lfloor n/2 \rfloor$

2.6 큰 정수의 계산법



▪ 자릿수가 분할된 두 정수의 곱셈

- 두 개의 정수 u, v를 분할하여 곱셈 연산을 함

$$u = x \times 10^m + y$$

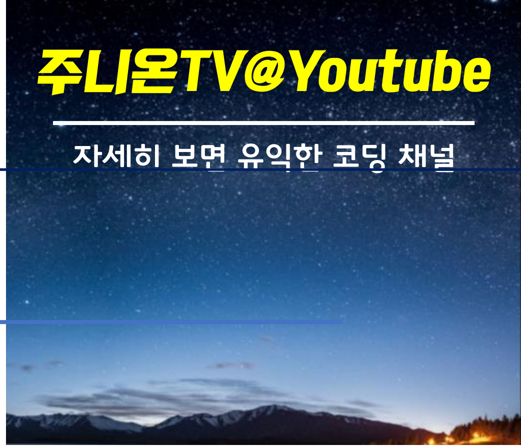
$$v = w \times 10^m + z$$

더 작은 자릿수의 곱셈으로 분할

$$\begin{aligned} uv &= (x \times 10^m + y)(w \times 10^m + z) \\ &= xw \times 10^{2m} + (xz + yw) \times 10^m + yz \end{aligned}$$

$$\begin{aligned} 567{,}832 \times 9{,}423{,}723 &= 567 \times 9{,}423 \times 10^6 \\ &+ (567 \times 723 + 9{,}423 \times 832) \times 10^3 + 832 \times 723 \end{aligned}$$

2.6 큰 정수의 계산법



Algorithm 2.9: Arithmetic for Large Integers

```
def prod (u, v):
    n = len(u) if (len(u) > len(v)) else len(v)
    if (len(u) == 0 or len(v) == 0):
        return [0]
    elif (n <= threshold):
        return lmult(u, v)
    else:
        m = n // 2
        x = div(u, m); y = rem(u, m)
        w = div(v, m); z = rem(v, m)
        p1 = prod(x, w)
        p2 = ladd(prod(x, z), prod(w, y))
        p3 = prod(y, z)
        return ladd(ladd(exp(p1, 2*m), exp(p2, m)), p3)
```

2.6 큰 정수의 계산법

- 큰 정수의 지수 곱셈과 나눗셈
  - 10의 지수 m으로 곱하기
    - 왼쪽으로 m 자릿수만큼 쉬프트
  - 10의 지수 m으로 나눈 나머지와 몫
    - 1의 자리에서 m의 자리까지가 나머지
    - $m+1$에서 n의 자리까지가 몫

$$567 \times 10^3 = 567{,}000$$

$$567{,}832 \text{ div } 10^3 = 567$$

$$567{,}832 \text{ rem } 10^3 = 832$$

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

Algorithm 2.9: Arithmetic for Large Integers

```
def exp (u, m):
    if (u == [0]):
        return [0]
    else:
        return ([0] * m) + u

def div (u, m):
    if (len(u) < m):
        u.append(0)
    return u[m : len(u)]

def rem (u, m):
    if (len(u) < m):
        u.append(0)
    return u[0 : m]
```

2.6 큰 정수의 계산법

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

```
u =  [2, 3, 8, 7, 6, 5]
v =  [3, 2, 7, 3, 2, 4, 9]
print(exp(u, 3))
print(div(u, 3))
print(rem(u, 3))
```

2.6 큰 정수의 계산법

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

- 임계값과 단순 곱셈
  - 임계값 (threshold): 특정 자리수까지 (threshold = 1)
  - 단순 곱셈: 전통적인 방법으로 곱셈

Algorithm 2.9: Arithmetic for Large Integers

```
def lmult (u, v):
    i = u[0] if (0 < len(u)) else 0
    j = v[0] if (0 < len(v)) else 0
    value = i * j
    carry = value // 10
    result = []
    result.append(value % 10)
    if (carry > 0):
        result.append(carry)
    return result
```

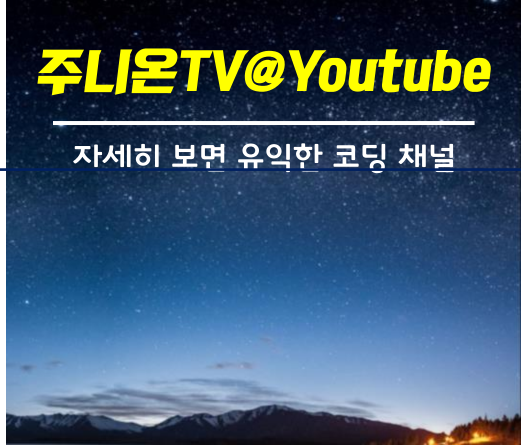
```
print(lmult([8], [7])[::-1])
print(8 * 7)
```

2.6 큰 정수의 계산법

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

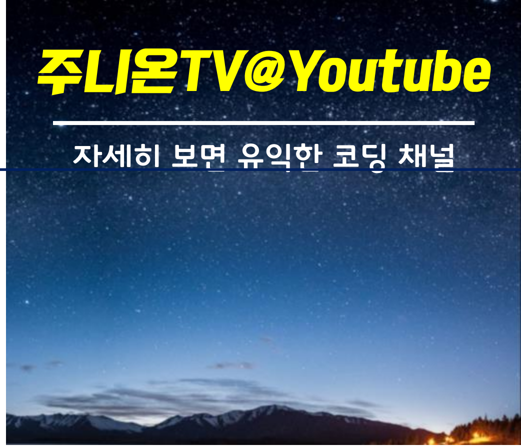
```
u =  [2, 3, 8, 7, 6, 5]
v =  [3, 2, 7, 3, 2, 4, 9]
print(567832 * 9423723)
print(prod(u, v)[::-1])
```

2.6 큰 정수의 계산법



- 큰 정수의 곱셈 알고리즘으로 우리가 한 일은?
  - 기본 연산: 한 자릿수에서의 단위 연산(총 m번 실행)
  - 입력 크기: 두 정수의 자릿수(n개의 자릿수)
  - 최선/최악/평균
    - 최악의 경우는 두 정수에 모두 0이 하나도 없을 때
  - Algorithm 2.9의 시간 복잡도 분석:
    - prod( ) 함수에서 재귀 호출을 네 번 한다는 것에 주목
    - $W(s) = 0, W(n) = 4W(n/2) + cn$
    - $W(n) \in \Theta(n^{\log_2 4}) = \Theta(n^2)$
  - 여긴 어디? 나는 누구? : 우리는 지금까지 뭘 한 걸까?

2.6 큰 정수의 곱셈



- Algorithm 2.9의 효율성 개선
  - 재귀 호출을 4번이나 하니까 효율성이 개선될 수 없다.
  - 재귀 호출의 횟수를 줄일 수는 없을까?

$$uv = xw \times 10^{2m} + (xz + yw) \times 10^{m} + yz$$

$$r = (x + y)(w + z) = xw + (xz + yw) + yz$$

$$r = (x + y)(w + z)$$
$$(xz + yw) = r - (xw + yz)$$

xw $\qquad$ $xz + yw$ $\qquad$ yz

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

Algorithm 2.10: Large Integer Multiplication 2 (Enhanced)

```
def prod2 (u, v):
    n = len(u) if (len(u) > len(v)) else len(v)
    if (len(u) == 0 or len(v) == 0):
        return [0]
    elif (n <= threshold):
        return lmult(u, v)
    else:
        m = n // 2
        x = div(u, m); y = rem(u, m)
        w = div(v, m); z = rem(v, m)
        r = prod2(ladd(x, y), ladd(w, z))
        p1 = prod2(x, w)
        p3 = prod2(y, z)
        p2 = lsub(r, ladd(p1, p3))
        return ladd(ladd(exp(p1, 2*m), exp(p2, m)), p3)
```

2.6 큰 정수의 곱셈

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

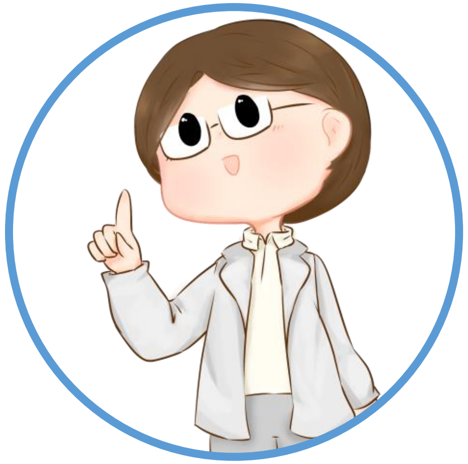
- prod2()의 시간 복잡도는?
  - 재귀 호출의 숫자를 **3회**로 줄임
  - $W(n) \in \Theta\left(n^{\log_2 3}\right) \approx \Theta(n^{1.58})$

```
u =  [2, 3, 8, 7, 6, 5]
v =  [3, 2, 7, 3, 2, 4, 9]
print(567832 * 9423723)
print(prod(u, v)[::-1])
print(prod2(u, v)[::-1])
```

2.6 큰 정수의 곱셈

주니온TV@Youtube
자세히 보면 유익한 코딩 채널

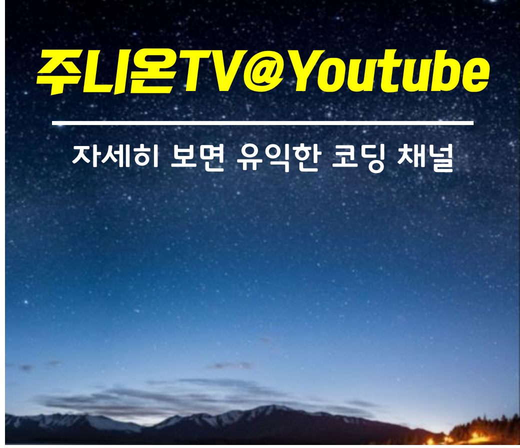
```python
def lsub (u, v):
    n = len(u) if (len(u) > len(v)) else len(v)
    result = []
    borrow = 0
    for k in range(n):
        i = u[k] if (k < len(u)) else 0
        j = v[k] if (k < len(v)) else 0
        value = i - j + borrow
        if (value < 0):
            value += 10
            borrow = -1
        else:
            borrow = 0
        result.append(value % 10)
    if (borrow < 0):
        print("음의 정수는 처리 못함.")
    return result
```



주니온TV@Youtube

자세히 보면 유익한 코딩 채널

https://bit.ly/2JXXGqz



- 여러분의 구독과 좋아요는 강의제작에 큰 힘이 됩니다.
- 강의자료 및 소스코드: 구글 드라이브에서 다운로드 (다운로드 주소는 영상 하단 설명란 참고)

https://bit.ly/3fN0q8t