20150390 서동민

전역함수: is_operator(token)
- 인풋 token이 operator(+ / - / *)인지 아닌지 반환합니다

전역함수: get_operator_priority(token)
- 인풋 token의 우선순위를 반환합니다 (+:1, -:1, *:2)

전역함수: tokenizer(str)
- 스트링을 토큰화된 어레이로 변경하여 리턴합니다

전역함수: check_has_invalid_token(str)
- 잘못된 문자(토큰, 스페이스가 아닌 문자)가 인풋에 포함되어있는지 체크합니다

클래스: ParsingTree

get_head()
- Tree의 가장 위(head)의 노드를 반환합니다

set_head(token)
- head에 token 정보를 가지고 있는 Node를 넣어줍니다

append_token(token)
- Tree의 마지막으로 추가된 노드 위치에 token을 넣어줍니다 (여기부터 시작하여, 자신의 노드 위치를 찾아갑니다)

print(leftward=optional)
- Tree 형태를 ouput형태로 프린트합니다

is_valid()
- Tree가 valid한지 체크합니다

클래스: Node

set_left(left), set_right(right)
- 해당노드의 자식을 셋해줍니다

is_head
- 해당 노드가 head인지 반환합니다

set_token
- 토큰을 해당 노드 위치에 삽입을 시도합니다
- 입력 토큰이 연산자가 아닐경우
    - 좌, 우측 자식이 모두 차있다면, 잘못된 인풋이므로 error를 raise합니다
    - 아니면, 해당 노드의 빈 자식에 해당 토큰을 가지고있는 노드를 추가합니다
- 입력 토큰이 연산자일경우
    - head일경우 head를 입력 토큰의 노드로 변환하고 자신을 자식으로 변경합니다
    - 아닐경우, 부모에게 입력 토큰을 flow_up함수를 이용하여 태웁니다

flow_up(token)
- 연산자의 우선순위를 비교하여 입력 토큰의 노드가 들어갈 위치를 결정하는 함수입니다
- flow_up은 무조건 자신의 토큰과 인풋 토큰이 연산자이여야합니다
- head일경우, head위치에 입력 토큰의 노드를 삽입합니다

- 입력 토큰의 우선순위가 높을 경우, 자신과 우측 자식 사이에 해당 토큰의 노드를 삽입합니다
- 입력 토큰의 우선순위가 같거나 낮을 경우, 부모노드에서 flow_up을 시도합니다

print()
- 노드의 자식들을 output 형태로 출력합니다

is_valid()
- 노드의 자식들이 valid한지 체크합니다
- 트리가 비어있지 않으며, 모든 리프가 아닌 토큰은 연산자이며 모든 리프는 연산자가 아니면 valid한 토큰 입니다

전체 플로우

1. 토큰화
- input에 스페이스를 제외한 토큰이 아닌 문자가 포함되어있으면 error를 raise합니다
- 정규식을 사용하여(tokenizer 함수) valid한 문자들만 추출하여 배열로 변경합니다

2. 트리구축
- 1.의 토큰들을 ParsingTree에 순서대로 하나씩 삽입(append_token) 하면서 트리를 구축합니다
- 삽입은 마지막으로 삽입한 노드에서부터 탐색하며 이루어지며, 연산자가 아니면 단순하게 자식으로 삽입, 연산자이면 재귀적으로 부모노드의 연산자와의 우선순위를 비교하며 자신의 노드 위치를 찾아 삽입합니다 (set_token, flow_up 함수 셜명 참조)

3. 프린트
- 트리를 좌측깊이우선탑색으로 순환하며 토큰을 프린트합니다
- 토큰화, 트리생성시에 error가 발생하거나 is_valid 함수가 false 를 반환하면 ‘incorrect syntax’를 프린트 합니다