

프로그래밍 언어 개론 00분반

201402447 한원희

Transition Matrix

	Digit				'-'	Alpha					
	'0'	'1'	...	'9'		'a'	...	'z'	'A'	...	'Z'
0	2	2	2	2	1	3	3	3	3	3	3
1	2	2	2	2	-1	-1	-1	-1	-1	-1	-1
2	2	2	2	2	-1	-1	-1	-1	-1	-1	-1
3	3	3	3	3	-1	3	3	3	3	3	3

```

, the values of the other entries are all 0.
for(int j = 0; j < 4; j++) {
    for(int i = 0; i < 128; i++) {

        if(j == 0) {
            if(i >= 48 && i <= 57)           // 아스키코드에서 숫자를 나타냄
                transM[j][i] = 2;
            else if(i == 45)                 // '-'
                transM[j][i] = 1;
            else if(i >= 65 && i <= 90)       // 'A' ~ 'Z'
                transM[j][i] = 3;
            else if(i >= 97 && i <= 122)     // 'a' ~ 'z'
                transM[j][i] = 3;
            else
                transM[j][i] = -1;
        }

        else if(j == 1 || j == 2) {        // 1일때와 2일때는 동일
            if(i >= 48 && i <= 57)           // 숫자
                transM[j][i] = 2;
            else
                transM[j][i] = -1;
        }

    }
}

```

InitTM은 하나씩 차례로 검사하며 조건에 충족한다면 값을 정해주는 형식 이니까 이중 for문에 if문으로 작성하였습니다.

```

private Token nextToken(){
    int stateOld = 0, stateNew;

    //토큰이 더 있는지 검사
    if(!st.hasMoreTokens()) return null;

    //그 다음 토큰을 받음
    String temp = st.nextToken();

    Token result = null;

    for(int i = 0; i<temp.length();i++){
        //문자열의 문자를 하나씩 가져와 현재상태와 TransM를 이용하여 다음상태를 판별

        //만약 입력된 문자의 상태가 reject 이면 에러메세지 출력 후 return함
        //새로 얻은 상태를 현재 상태로 저장
        stateNew = transM[stateOld][temp.charAt(i)];    // charAt을 이용해 문자를 판별

        if(stateNew == -1){    // state가 -1이라면 잘못된 값
            System.out.println("Error.");
            return result;    // null을 리턴 해줍니다
        }

        stateOld = stateNew;    // 새로 얻은 상태를 현재 상태로
    }
}

```

nextToken 내부에 있는 for문입니다.

설명에 써 있는대로 작성하였습니다. 위에 선언된 stateOld와 stateNew를 사용해 줍니다. stateNew로 다음 상태를 판별하기 위한 변수로 사용하고 stateOld는 현재 검사하는 단계로 봅니다. stateNew가 -1 이라면 for문 바로 위에 선언돼 있는 result 즉, null을 리턴해주고 새로 얻은 상태를 현재 상태로 바꾸어줍니다.

```

public List<Token> tokenize() {
    //입력으로 들어온 모든 token에 대해
    //nextToken()이용해 식별한 후 list에 추가해 반환

    List<Token> tokens = new ArrayList<Token>();    // 새로운 tokens list 생성

    Token next = this.nextToken();

    while(next != null){    // null이 아닐때까지 진행해 줍니다

        tokens.add(next);    // null이 아니라면 현재 토큰을 list에 추가
        next = this.nextToken();    // 다음 토큰으로 넘어갑니다

    }

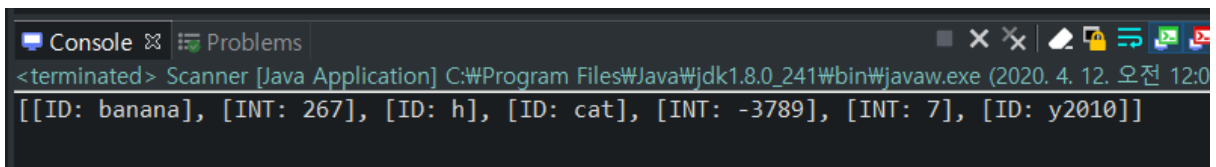
    return tokens;    // list 반환
}

```

설명을 먼저 본다면 모든 token에 대해 nextToken()을 이용해 식별한 후 list에 추가해 반환 이라고 적혀 있습니다.

그러면 필요한 것은 새로운 token list와 토큰을 저장해 줄 token 변수 입니다. 둘을 선언해 준 뒤에 토큰들을 하나씩 검사해 주어야 하므로 while(next != null) 로 null이 나올때까지 계속해서 검사해줍니다.

List에 token을 add해주고 다음 토큰으로 진행해줍니다.



```

<terminated> Scanner [Java Application] C:\Program Files\Java\jdk1.8.0_241\bin\javaw.exe (2020. 4. 12. 오전 12:0)
[[ID: banana], [INT: 267], [ID: h], [ID: cat], [INT: -3789], [INT: 7], [ID: y2010]]

```

실행 결과입니다