프로그래밍언어 개론 보고서

[04] Recognizing Token Switch Case

제출일	2017/06/29
학 번	201000287
소 속	일어일문학과
이 름	유다훈

1 과제문제

1 주어진 코드에서 함수 구현

- nextToken() : 주어진 문자열을 검사하여 ID 혹은 INT 타입의 타입으로 분류
- tokenize() : 주어진 문자열을 토큰화시키는 메소드

- 과제의 목적
  - 특정 문자열을 입력 받을 경우, 해당 문자열을 나누어서 각 요소를 인식하여 속성 대로 분류하고 출력해주는 Scanner
  - 저번 과제와는 달리 Switch-Case 문법을 이용하여 구현

2 과제 해결 방법

1 nextToken()

- 전달받은 문자열에서 문자를 하나씩 뽑아, Switch 문을 통하여 문자의 상태가 문자형태인지, 숫자형태인지를 파악한다.
- 검사가 끝난 후에는 해당 문자열의 타입을 확인하여 문자열을 ID 혹은 INT 형태의 토큰으로 분류하고 저장한다.

```
private Token nextToken() {
    int state = 0;
    boolean errorState = false;
    // 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() && !errorState; i++) {
        switch (state) {
        case 0:
            if (Character.isDigit(temp.charAt(i)))
                state = 2;
            else if (temp.charAt(i) == '-')
                state = 1;
            else if (Character.isLetter(temp.charAt(i)))
                state = 3;
            else
                errorState = true;
            break;
        case 1:
            //상태에 따라state를 변경
            if (Character.isDigit(temp.charAt(i)))
                state = 2;
            break;

        case 2:
            //상태에 따라state를 변경
            if (Character.isDigit(temp.charAt(i)))
                state = 2;
            break;
        case 3:
            //상태에 따라state를 변경
            if (Character.isDigit(temp.charAt(i)))
                state = 3;
            else if (Character.isLetter(temp.charAt(i)))
                state = 3;
            break;
```

```
        default:
            System.out.println("Case error: " + temp);
            return result;
        }

    }
    if (errorState) {
        System.out.println("acceptState error: " + temp);
        return result;
    }

    switch (state) {
    case 2: //int
        result = new Token(TokenType.INT, temp);
        break;
    case 3: //id
        result = new Token(TokenType.ID, temp);
        break;
    }
    return result;
}
```

2 tokenize()

- while 문을 이용하여 입력받을 토큰이 없을 때까지 토큰분류하는 검사문을 실행한다.

```
public List<Token> tokenize() {
    List<Token> tokens = new ArrayList<Token>();
    Token t = null;

    //��ũ List�� ��ó��ϴ��� ��ȭ
    while (true) {
        t = nextToken();

        if(t == null)
            break;

        tokens.add(t);
    }


    return tokens;
}
```

3 과제를 해결하면서 느낀 점

1. 이번 과제는 Switch 문을 활용하여 구현해보았는데, 저번 과제를 벌써 수행해서 그런 것인지는 모르겠으나 프로그램 이해하기가 훨씬 수월하였다.
2. 이전 과제보다 Switch 문을 통하여 토큰검사하는 알고리즘을 작성하기가 훨씬 쉬웠다.