PL Assignment #3 : Recognizing Tokens

과제물 부과일 : 2019-03-20(수)

Program Upload 마감일 : 2019-04-04(목) 23:59:59

문제

다양한 형태의 identifier, integer number(**음수 포함**) 들로 이루어진 text file을 입력 받아, 각 요소를 인식하여 출력하는 program을 작성하시오. Input file name은 as03.txt이다.

예를 들어 as03.txt file의 내용이 아래와 같다면,

```
banana   267  h  cat  -3789  7   y2010
```

출력은 아래와 같아야 한다.

```
id: banana
int: 267
id: h
id: cat
int: -3789
int: 7
id: y2010
```

작성해야 할 부분

Transaction Metrix 완성 (아래 mDFA를 나타내는 TM작성)

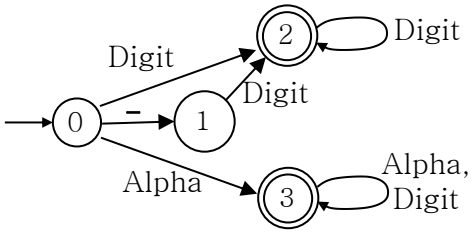
`nextToken()`함수 작성

`tokenize`함수 작성

Regular Expression

```
id:     Alpha[Alpha|Digit]*
int:    Digit+ | "-" Digit+
Alpha:  [A-Z] | [a-z]
Digit:  [0-9]
```

mDFA



위의 DFA 를 Transaction Metrix 로 표현한 일부는 아래와 같다.

	Digit				'-'	alpha					
	'0'	'1'	…	'9'		'a'	…	'z'	'A'	…	'Z'
0	2	2	2	2	1	3	3	3	3	3	3
1	…	…	…	…	…	…	…	…	…	…	…
2	…	…	…	…	…	…	…	…	…	…	…
3	…	…	…	…	…	…	…	…	…	…	…

Programming

Token 표현하기

```
public enum TokenType{
	ID(3), INT(2);

	private final int finalState;

	TokenType(int finalState) {
		this.finalState = finalState;
	}
}
```

Data Type

```
public static class Token {
	public final TokenType type;
	public final String lexme;

	Token(TokenType type, String lexme) {
		this.type = type;
		this.lexme = lexme;
	}

	@Override
	public String toString() {
		return String.format("[%s: %s]", type.toString(), lexme);
	}
}
```

Programming-TM

Control state transition of transition diagram by TM(transition matrix).

"Array of accept(final) states"를 사용

```
private int transM[][];
private String source;
private StringTokenier st;


public Scanner(String source) {
	this.transM = new int[4][128];
```

```
        this.source = source == null ? "" : source;
        this.st = new StringTokenizer(this.source, " ");
        initTM();
}

private void initTM() {
      // transM[4][128] = { {...}, {...}, {...}, {...} };
      // values of entries:  -1, 0, 1, 2, 3 : next state
      // TransM[0]['0'] = 2, ..., TransM[0]['9'] = 2,
      // TransM[0]['-'] = 1,
      // TransM[0]['a'] = 3, ..., TransM[0]['z'] = 3,
      // TransM[1]['0'] = 2, ..., TransM[1]['9'] = 2,
      // TransM[2]['0'] = 2, ..., TransM[1]['9'] = 2,
      // TransM[3]['A'] = 3, ..., TransM[3]['Z'] = 3,
      // TransM[3]['a'] = 3, ..., TransM[3]['z'] = 3,
      // TransM[3]['0'] = 3, ..., TransM[3]['9'] = 3,
      // ...
      //    The values of the other entries are all -1.
}
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 다음
        상태를 판별
            //만약 입력된 문자의 상태가 reject 이면 에러메세지 출력 후 return함
            //새로 얻은 상태를 현재 상태로 저장
        }
        for (TokenType t : TokenType.values()){
                if(t.finalState == stateOld){
                        result = new Token(t, temp);
                        break;
                }
        }
        return result;
}

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

        return …
}

public static void main(String[] args){
        FileReader fr = new FileReader("c:/as03.txt");
        BufferedReader br = new BufferedReader(fr);
        String source = br.readLine();
        Scanner s = new Scanner(source);
        List<Token> tokens = s.tokenize();
        System.out.println(tokens);
}
```

유의 사항

- 입력 data는 프로그램을 제대로 검증할 수 있는 data로 구성되어야 한다.
- 반드시 transition matrix를 사용하여야 한다.
- 위에 나타난 코드는 한 class 내에 작성하여야 한다.
- 기타 과제 제출에 관한 구체적인 제반 사항은 각 TA의 지침에 따른다.

수정: 2019-03-19