Design & Report for CS 3361 Project 1

By Miguel Cruz, Minoshun Renganathan and Isman Daiyrov

1. Introduction

- In this project we are using Java language that has a scan function that scans an input file for valid and invalid tokens. If the token is valid the scan function returns the token type, otherwise returns error flag.

2. Data Structure

- *tokenType*: an array. For any final state *s*, *tokenType[s]* is the token type concludable from final state *s*
- *"error" token*: when there is an error in an input, the token will be produced by scan(...) function
- *transitionTable*: a 2-dimensional array. The 1st dimension is indexed from 1-18 (the number of states), and the 2nd dimension is indexed from 1-14 with representing a current input character
- *transitionTable[i][j]* (for any integer *i* and *j*): a record with field names
  - i. *action* and *newState*.
  - ii. *action* can take values of *move*, *recognize*, *stuck*.

<u>Example</u>:

- *transitionTable[i][j].action = move:* the automata should move to next state
- *transitionTable[i][j].newState:* the state where to move to
- *transitionTable[i][j].move = recognize:* means that *i* is a final state *s* and the automata cannot move to any other state with the input character corresponding to the number *j*, so then we recognize a token
- *transitionTable[i][j].move = stuck*: means the automata cannot get to any state from state *i* with a character corresponding to the number *j*

3. Algorithms (in pseudocode)

```
File foo.txt
ReadFile = open(“name.txt”, a)

current_state = 0

for(int pointer = 0; pointer <= file.length(); pointer++){

    current_character = pointer.file

    if (ReadFile != NULL) {
        if(current_character == “ ” | “\t” | “\n”)
            current_state = 1    //start
            continue

    if(current_character == ‘/’)
        current_state = 2    //div
        System.out.println(“div”)
        continue

        if(current_character == ‘/’)
            current_state = 3    //new line / non-newline
            continue to state_1 //start

        else if(current_character == ‘*’)
            current_state = 4    //non-*
            continue

            if(current_character == ‘*’)
                current_state = 5    //non-/or*
            else if(current_character == ‘/’)
                current_state = 1
            current_state = 1    //start
    continue

    if(current_character == ‘(‘)
        current_state = 6    //lparen
        System.out.println(“lparen”)
        continue

    if(current_character == ‘)’
        current_state = 7    //rparen
        System.out.println(“rparen”)
        continue
```

```
if(current_character == ‘+’)
        current_state = 8    //plus
        System.out.println(“plus”)
        continue

if(current_character == ‘-’)
        current_state = 9    //minus
        System.out.println(“minus”)
        continue

if(current_character == ‘*’)
        current_state = 10  //times
        System.out.println(“times”)
        continue

if(current_character == ‘:’)
        current_state = 11
        System.out.println(“semicolon”)
        continue
        if(current_character == ‘=’)
                current_state = 12  //assign
                System.out.println(“assign”)
                continue

if(current_character == ‘.’)
        current_state = 13
        System.out.println(“period”)
        continue
        if(current_character == digit)
                current_state = 15 //number
                if(digit digit^)
                        current_state = 15  //number

if(current_character == digit)
        current_state = 14  //number
        continue
        if(digit digit^)
                current_state = 15  //number
        System.out.println(“number”)

if(current_character == letter)
        current_state = 16  //id or keyword
        continue
        if(letter letter|digit^)
                current_state = 16  //id or keyword
        System.out.println(“id”)
        //use java utilities to compare “Characters”

    if((letter letter^) == “read”)
        System.out.println(“read”)

    if((letter letter^) == “write”)
        System.out.println(“write”)

    else {
        System.out.println(“ERROR”)
    }

    add to tokenArray
    print tokenArray
}
```

4. Test Cases

- **Example Input File: example1.txt**

  ```
  read
  forty 54
  /*
  ```

  Example Output = (read, id, number, div, times)

- **Example Input File: example2.txt**

  ```
  /*this is a test comment*/
  55
  variable +
  ```

  Example Output = (number, id, plus)

Acknowledgement

Thanks Peter Adamopoulos and Zanxiang Wang for useful discussions.

Created by YL March 2021