**LFCD Lab3 Documentation**

Bogdan Cristina

Git link: <https://github.com/bcie2480/LFCD>

In the class “Scanner” I have:

self.\_\_symbolTable = SymbolTable(10) # the Symbol table of the program (output)

self.\_\_programInternalForm = {} #the PIF (output)

self.\_\_listOfTokens = [] #an auxiliary list of tokens

self.readFromFile("tokens.txt") #this function fills the list of tokens with values

#from my file of tokens(tokens.txt)

self.\_\_input = "" #the program to be executed

self.readProgram(problem) # this function reads the program from file and assigns to the input attribute a string with the program body

In order to perform the scan I have the following functions:

**scan()**

Input: - Output: PIF + ST (message in case of lexical errors)

The main function, performs the splitting of tokens and then the classification

The splitting of input is performed using the functions:

**splitInputSeparators()**

Input: -

Output: separatedInput (a list)

Splits a program for all separator characters

**splitInputOperators()**

Input: -

Output: allTokens (a list)

Splits an instruction (that was previously splited for separators) for all operator characters

Auxiliary functions for splitting functions:

**isSeparator(char)**

Input: char (a single letter string)

Output: True if char is in the list of separators, false otherwise

**isOperator (char)**

Input: char (a single letter string)

Output: True if char is in the list ofoperators, false otherwise

For the classification of tokens I used the function:

**clasifyToken(token)**

Input: token (a string)

Output: - (raises exception if needed)

First performs a strip to avoid white spaces at the beginning/end of token. Then it clasifies the tokens by the following rules:

If the token is in the list of tokens (in the file tokens.txt) it is placed in PIF, with the id -1

If the token is a valid identifier and it’s not in ST, it is placed in ST, and then added in PIF as a pair of “id” and its id from ST

If the token is a valid identifier and it already exists in ST, it is just added in PIF as a pair of “id” and its id from ST

If the token is a valid constant it is added in PIF as a pair of “const” and its value

Auxiliary functions for classifying:

**isValidIdentifier(token)**

Input: token (a string)

Output: True if token is a valid identifier(starts with letter and doesn’t contain invalid characters), false otherwise

**isValidConstantInt(token)**

Input: token (a string)

Output: True if token is a valid constant int(contains only digits, first digit is not 0), false otherwise

**isValidConstantString(token)**

Input: token (a string)

Output: True if token is a valid constant string(starts and ends with ‘ and doesn’t contain forbidden characters), false otherwise