FLCD SCANNER DOCUMENTATION

TONCEA ION-ALIN 937/2

Link to git: https://github.com/TonceaAlin/FLCDlab

Requirements

Implement a scanner (lexical analyzer): Implement the scanning algorithm and use ST from lab 2 for the symbol table.

Input: Programs p1/p2/p3/p1err and token.in

Output: PIF.out, ST.out, message “lexically correct” or “lexical error + location”

ST

The symbol table is implemented using a List of List with Strings. In the SymbolTable class we just use the last-time implemented SymbolTable for keeping track of identifiers and constants.

HashFunction – ascii sum of the chars **mod** the size of the key

Other methods:

-add(key)

-search(key)

-findElementPosition(element)

-get

PIF

The program internal form represents a Lists of Pairs(which is implemented as a pair of generic elements). In our PIF we are talking about Pairs of Strings and Integers, each representing the key we are talking about and its position.

Tokenizing

We create tokens by using StringTokenizer class. We use as delimitators the list of separators. After that we go token by token in the resulting list and check if there are operators left in the content of each token.

Scanning

Using the list of tokens we check each time the case (identifiers, constants – ST or reservedWords, operators, separators – PIF) and we add it into the PIF with the corresponding position.