Link to github: https://github.com/eduard-C0/FLCD/tree/Lab3

PifEntry

This class has two attributes. One of them is the given token and the second one is his position.

getToken() method returns the token
getPosition() method returns the position
setToken() method set the token for the corresponding PifEntry
setPosition() method set the position for the corresponding PifEntry

Pif

This class has a list of entries of type PifEntry.

writeToPifFile() method writes each entry from the entries list to the Pif.out file
addElement() method adds an element to the entries list

ScannerFile

This class has as attributes a list of acceptedTokens, a file name representing the path from which we are getting the tokens, a symbol table, a program information file and a file name representing the path from which we are getting t he program we want to parse

readFromTokenFile() this method reads line by line all the tokens from the corresponding file and will remove all th e spaces
IO errors will be caught and printed

readFromProgramFile() this method reads line by line from the program file and will try to parse it calling the parse Line function.
If a lexical error will be thrown by the parseLine function we will print into files what we had until then into the Pif and Symbol table, we will close the file from which we are reading and we will print the lexical error.
If there was no lexical error at the end we will write into files the Pif and the Symbol Table.

parseLine() this method will split the line of code ,given as parameter, by space, we will store them into a queue and will iterate through all the elements analyzing them and classifying them as token, integer, string or identifier with t he help of the regex patterns.
If the element couldn't be classified by one of the above types it means that it is an composed element and we will it erate through all of our separators and operators and if they are contained we will split by them and add the result to the queue of elements.

RegexPatterns
- identifier "[a-zA-z][0-9a-zA-Z]*" means that we will take all the elements starting with a lower case or upper case l etter and followed by 0 or multiple digits(from 0 to 9)/lower case letters/ upper case letters
- integer "0|(-?[1-9][0-9]*)" means that we will take as an element 0 or any number starting with a digit from 1 to 9 f ollowed by 0 or multiple digits from 0 to 9 optionally having the '-' sign at the beginning
- string "\"([0-9a-zA-Z])+\"" means that we will take as an element starting with " followed by 1 ore more digits fro m 0 to 9 or any lower case or upper case letter and ending with another "

splitBySpace() this method will return a list of elements which are the results of splitting the given line by space

Link to github: https://github.com/eduard-C0/FLCD/tree/Lab2

SymbolTable represented as a alphabetically BinarySearchTree

```
    /**
     * addAndReturnPosition method returns a position of a token or if the token doesn't exist in the symbol table
     * that token will be added and the position will be returned.
     * @param key which is the token
     * @return position of the token
     */

    /**
     * add method adds a token to the symbol table.
     * @param key the token we want to add
     * @param value the position of the token
     */


    /**
     * search method searches for a token in the symbol table.
     *
     * @param key the token for which we are looking for
     * @return the position of the token
     */

    /**
     * inorderTraversal method goes through the symbol table represented as a binary search tree like this (Left, Root,
Right)
     *
     * @param currentNode starting node from where we want to start the inorder traversal
     * @param queue an empty queue for storing the order in which the nodes were visited
     */

    /**
     * nodesInAsscendingOrder method reorders the tokens alphabetically
     */
```