Lab 3 - Documentation

Github repository: https://github.com/alinaiovan/LFTC/tree/Lab3

Scanner

Symbol Table

I had to implement 2 symbol tables, one for the constants and one for de identifiers.

For the representation I used a custom Hash Table that has as methods the following:

- hash (that generates and returns the new hashed position for a new element that will be added),
- contains(a function that verifies if the hashtable contains a certain key),
- remove(removes a key), add(appends a new key to the generated hash pos),
- getPos(returns the position of a key as a tuple)

The symbol table is a wrapper for the hashmap.

The Scanner generates tokens that represent either a reserved word, a separator, an operator, an identifier or a constant,

searching char by char, not separating by space. For every char is checking the following situations:

- if it is part of an operator(in that case it will call another function which will return the hole operator) and adds it to the list of tokens
- if it is the separator ~ which denotes the starting of a string and calls another function that returns the whole string (until the next ~) and adds it to the list of tokens
- if it is a separator and adds it to the list of tokens
- if none of the above start creating a token and when one of the cases above is met, the token is added to the tokens array

PIF:

Represented using a custom Linked List and having the following mthods:

- add(token, posToken, pos) where token is th token itself(separator, operator, const,..), posToken is the token's position in the tokens table(their code) and pos is the token's position in the symbol table and adds them to the list

Main:

The tokens are taken one by one, if identifiers or constants they are added to their symbol table but also to the pif list and if reserved words, separators or operators they are only added to de pif list. If not, an error message is printed with the token and the line at which it occurred.