Documentation - lab3

https://github.com/iuliaaai/LFTC/tree/master/lab2

I implemented the SymbolTable using a HashTable as the data structure and it contains both the identifiers and the constants. The HashTable is composed of a list of lists, having the hash function computed as: for each string, the sum of the ascii code for each character % size of the table.

The tokenizing function reads char by char on each line and checks if we have a separator, the beginning of a string constant/char constant/identifier/constant, a part of an operator or a reserved word and appends these tokens to a list.

The scanning algorithm splits each line into tokens and adds the constants and identifiers to the SymbolTable or returns an error in case of a lexical error.

The program internal form is list of pairs, each pair having as the first element the token and as the second another pair containing the position in the symbol table and the code of the token: **(token, ((lineST, colST), tokenCode))**. After reading the file “token.in”, a code is assigned for each token (for constants the code is 0 and for identifiers the code is 1). In case of operators, separators and reserved words the position is (-1, -1), as they are not stored in the symbol table.