Laboratory 2

- Antonio Suciu, 937/1 -

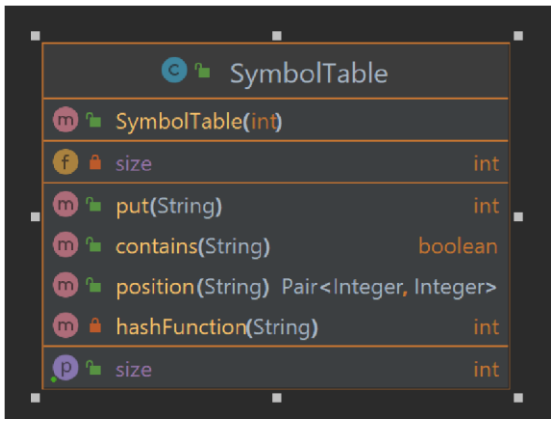
Statement: Implement the Symbol Table (ST) as the specified data structure, with the corresponding operations.

Github Link here

The assignment is the following:

1. Symbol Table:
   b. separate tables for identifiers, respectively
2. Symbol Table (you need to implement the data structure and required operations):
   d. hash table

The representation of this Symbol Table is a hash table.
The hash function is computed by summing up all of the ascii codes of the characters of the key, modulo the length of the table.
When two elements have the same hash value, they will be both added to the corresponding array (of the said hash value).



Operations: put(adds a key to the ST), contains(checks whether the given key is contained), position(gives the exact position of a given key)

Laboratory 3

- Antonio Suciu, 937/1 –

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

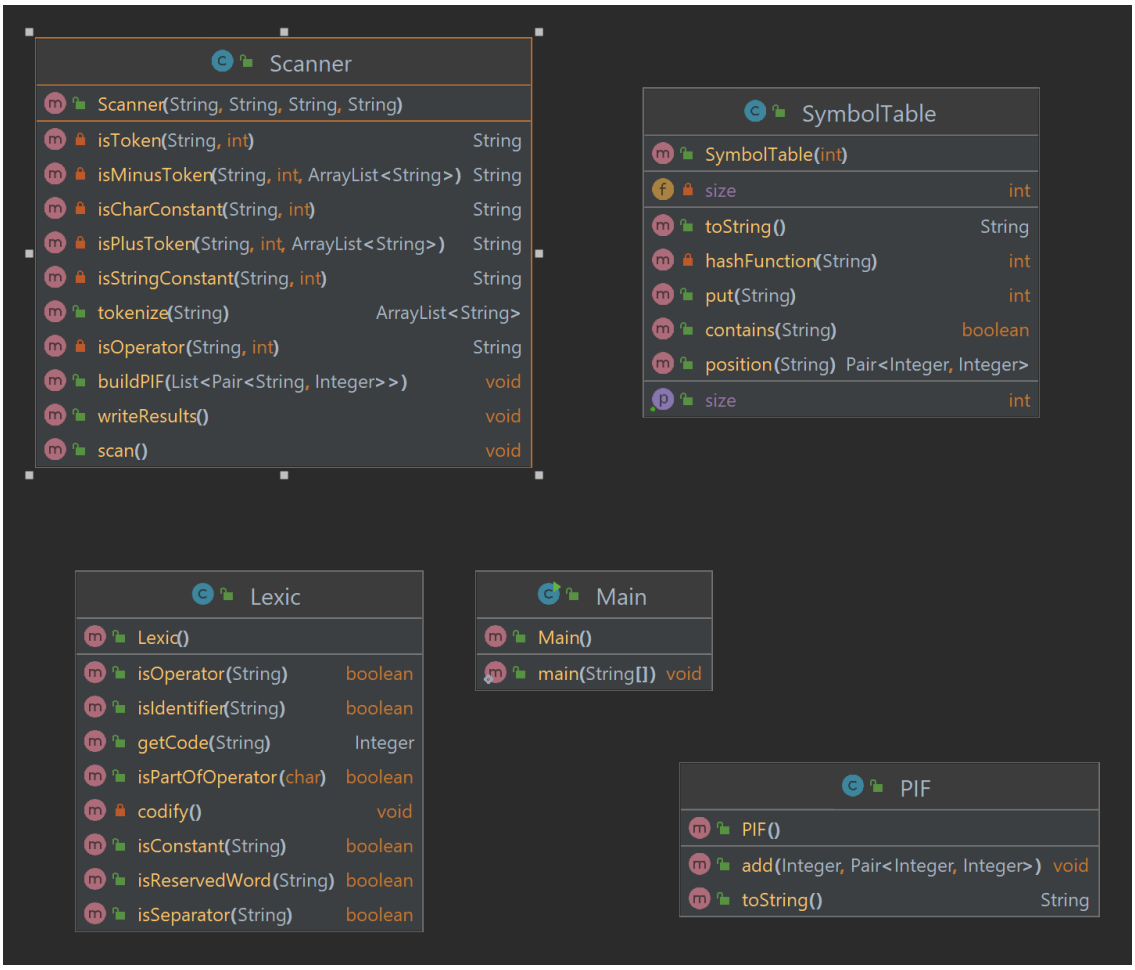
Input: Programs p1/p2/p3/p1err and token.in (see Lab 1a)

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

Deliverables: input, output, source code, documentation

Details:

- ST.out should give information about the data structure used in representation
- If there exists an error the program should give a description and the location (line and token)



PIF (program internal form)
The PIF is represented as a list of Pairs.
The first element of the said pairs is the code of the token, while the second element is another pair with the position in the corresponding symbol table (constant/identifier)

Scanner
The scanner takes each line token by token, in order to check whether it is an operator, separator, reserved word, or a constant / identifier (it also does the look-ahead bit when it comes to composed operators). Then, based on the list of all the tokens, we build the pif either like ASCII CODE -> (-1, -1) or CODE_FOR_SOMETHING -> (ACTUAL HASH, ACTUAL POSITION RELATED TO THE HASH)
If the token is none of the above, we have an error and we keep track of it

```
String numconstRegex = "^0|[+|-][1-9](\\d)*|[1-9](\\d)*|[+|-][1-9](\\d)*$";
/// zero | sign nzd {zd} | nzd {zd} |
String charconstRegex = "^'[a-zA-Z0-9_?!#*./%+=<>;)(){ ]'";
String stringconstRegex = "^\"[a-zA-Z0-9_?!#*./%+=<>;)(){ ]+\"";
```