Formal Languages and Compiler Design – Lab 3

https://github.com/baraganio/Formal-languages-and-compiler-design/tree/master/Lab%203

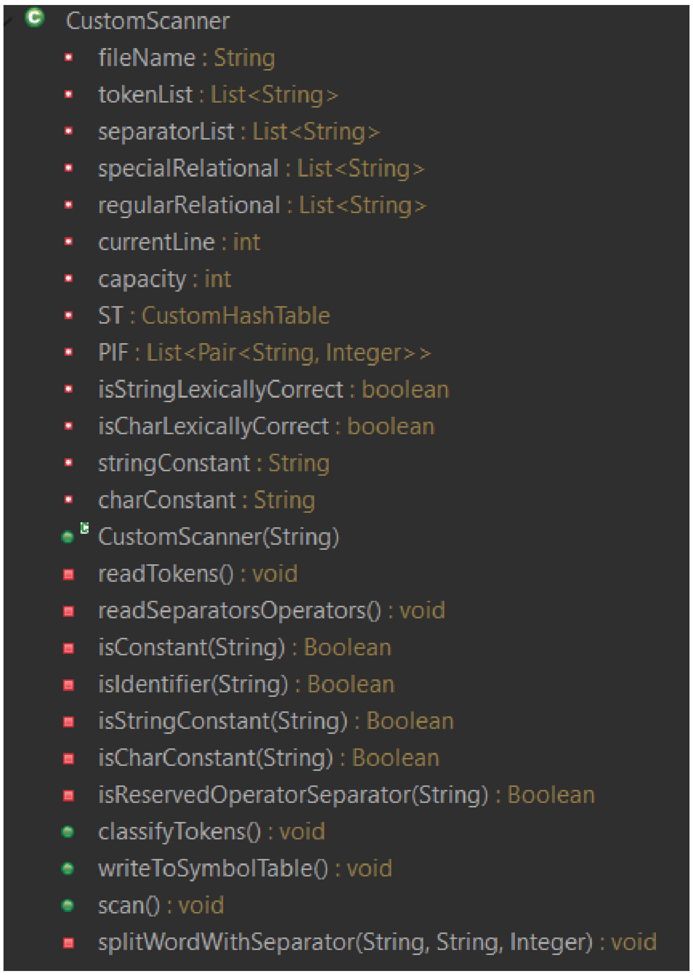
The project consists of three java classes and the Main class. The last one is only used to launch the scanner.

One of the others is the Pair class which only has two properties, key and value. It will be used to store in the PIF the token (key) and its position on the Symbol Table (value).

The CustomHashTable class is the same as previous laboratory (lab2). For the hashFunction I calculate the HashCode of the value I am going to store and divide it by the size of the HashTable. Then I take the reminder of the division and in that position will be the value stored. If two values have the same position, I will repeat the hashFunction for the second value to store, but this time adding one to the HashCode before dividing it by the HashTable size. The CustomHashTable class represents the data structure itself and has as attributes an String array and the number of elements it has stored. As methods, it have getters, constructor, insert, findPosition (which calculate the HashValue taking into account if there has been collisions or not), find (given the value, returns its position in the HashTable), toString, fHash (make the operations to calculate the HashValue), isPositivePrime (check if a number is prime, in order to use it to calculate the HashTable size) and nextPrimeNumber (it calculate the most effective size of the HashTable starting from a given size).

Finally, there is the CustomScanner class. It is the one in charge of read the input file and classify every token in the file. As principal properties it has token list (where will be stored each token red from the input file), separator list (there will be stored each separator red from the token.in file), special and regular relational lists (introduced manually) and obviously the Symbol Table (ST) and the PIF.

As methods there are: readTokens() in charge of read the tokens and reserved words on the token.in file and store them on the tokenList, readSeperatorsOperators() read the operator and separator tokens from the token.in file and store them in the separatorList(), classifyTokens() classify the tokens into reserved operators, identifiers and constant and insert the last two into de SymbolTable, also it prints the PIF and if there has been any lexical error. Scan() in charge of read every word on the input file and stored it and in what line it is, writeToSymbolTable() it prints to the st.out file the ST, splitWordWithSeparator() is in charge of divide words that have a separator like: a>=b it will be divided into a, >= and b.



```
CustomScanner
    fileName : String
    tokenList : List<String>
    separatorList : List<String>
    specialRelational : List<String>
    regularRelational : List<String>
    currentLine : int
    capacity : int
    ST : CustomHashTable
    PIF : List<Pair<String, Integer>>
    isStringLexicallyCorrect : boolean
    isCharLexicallyCorrect : boolean
    stringConstant : String
    charConstant : String
    CustomScanner(String)
    readTokens() : void
    readSeparatorsOperators() : void
    isConstant(String) : Boolean
    isIdentifier(String) : Boolean
    isStringConstant(String) : Boolean
    isCharConstant(String) : Boolean
    isReservedOperatorSeparator(String) : Boolean
    classifyTokens() : void
    writeToSymbolTable() : void
    scan() : void
    splitWordWithSeparator(String, String, Integer) : void
```