Github: https://github.com/915-Narita-Andrei/FLCD-lab

Symbol Table

This was implemented using as data structure a Hash Table. The Hash Table is implemented using a list which will have on every positon a list of strings. This data structure has an “add” method, which based on a string will compute a hash value which will be used as a key and store on the list on that key-position. If multiple strings will lead to the same hash values, there is no problem since on each position from the list there is another list which can keep more than one element. And this is the way how collisions are solved.

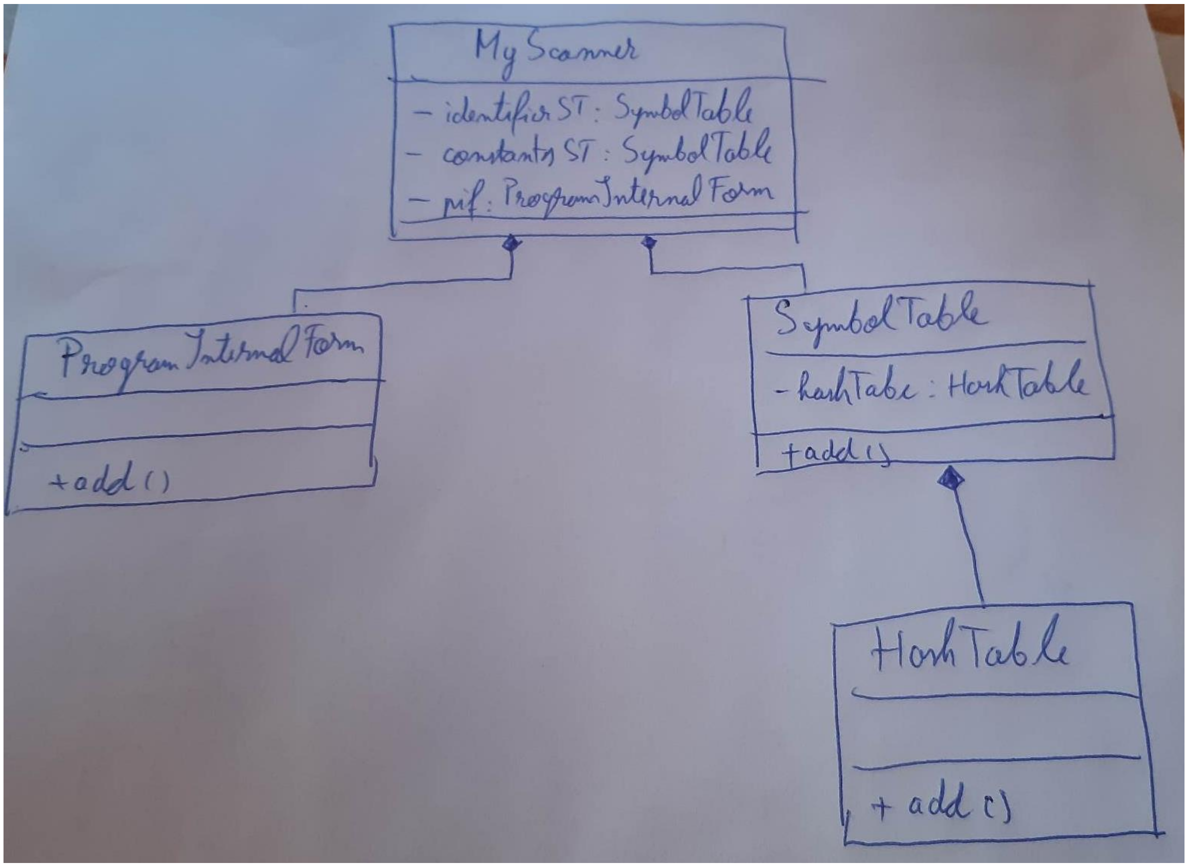
The hash function has the following computations based on a string: It will sum up all the ascii codes of the characters from the string. The result will be the previous sum modulo size of the list of elements. We also prefer to have the size as a prime number in order to avoid hash collisions.

Program Internal Form

Program Internal Form is a list of pairs of the form <String, Pair<Integer, Integer>> which will keep for every token from an input program a pair which corresponds with the value from Symbol table if the token is an identifier or a constant, or a simple pair consisting of (-1 -1).

Scanner

The scanner will read a program and based on a Token.in file (contain all separators, operators and reserved words) will split the program into token with the help of separators. Then for every token, we check if it is a separator, operator, reserved word, identifier or constant. For identifier we have a regex that will match every string that start with a letter and after can have zero or more letters and digits. For string constants we check to start and end with “ and then have any number of letters or digits. For character constants we check to start and end with ‘ and have exactly one letter or digit. For number constant we check just to be a single digit, or if more digits, then the first digit to be different then 0.


MyScanner
- identifierST : SymbolTable
- constantsST : SymbolTable
- pif : ProgramInternalForm
ProgramInternalForm
+add()
SymbolTable
-hashTabe : HashTable
+add()
HashTable
+ add ()