GitHub Link: https://github.com/pauladam2001/Sem5_FormalLanguagesAndCompilerDesign/tree/master/Lab3/Lab3-SymbolTable

I used a Hash Table for the implementation of the Symbol Table (unique for identifiers and constants). I used an Array of Pairs for solving collisions. After the hash function is called
for an element, if that positions already has an element, I add a new pair of the form Pair <String, Integer> (String being the element and Integer a unique counter incremented after each
added element).

For the Symbol Table I have 3 functions:
- getHash - hashes an element and returns the hash % size of the array
- add - adds an element to the symbol table. First we compute its hash, after we check if the element is already present in the symbol table (using the find function). If it's not, we add
   the element in form of a Pair<Element, Counter> and we increase the counter
- find - checks if an element is present in the symbol table. First we compute its hash, then, on the hash position we go through the pairs. If nothing is found the function returns true,
   or false otherwise

(I also have the Scanner implemented (without the Program Internal Form), but I saw that the Scanner is needed only
 for the next lab)