CS 4323
PROGRAMMING ASSIGNMENT #1
(SCANNER)
Due: March 3, 2022

This project builds the first component of a compiler, the lexical analyzer (or scanner), for a programming language called Simple-Scala. Write a program (in Java or any programming language of your choice) with subprocedures/classes SCANNER(), BOOKKEEPER() and ERRORHANDLER().

Construct SCANNER() from a DFA accepting the following four types of tokens:

- *Keywords*: package, import, abstract, final, sealed, private, protected,class, object, val, def, <=, if, else, while, case, =>, in, print, return, not, true, false, and, or, int, real, bool.
- *Identifiers*: any nonempty string of English letters, digits, and/or periods that begins with a letter and is not a keyword.
- *Constants*: any nonempty string of digits, optionally with a decimal point.
- *Special symbols*: #, ;, *{*, *}*, (,), :, ,, =, +, *, @.

A blank (many consecutive blanks are the same as a single blank), line break, or special symbol separates two tokens. Statements following # are comments (the line break ends the comment line); # and other words in comments are not considered tokens and should be ignored by SCANNER().

Call SCANNER() from the main body of the program, once for each token to be recognized, until all symbols in the given input program are consumed. Thus, the main body will contain a loop in which SCANNER() is called repeatedly, until all symbols of the source program are consumed. The special symbol $ indicates the end of the source program and is not a part of the source program.

Call BOOKKEEPER() from SCANNER() when an identifier or constant is recognized. It is responsible for maintaining a symbol table SYMTAB (of size 100) to store tokens passed from SCANNER() and their attributes, i.e., classification as to identifier or constant. Each identifier or constant must appear exactly once in the SYMTAB.

Call ERRORHANDLER() from SCANNER() when an illegal token is recognized. It is responsible for producing appropriate error messages together with the line number. Include at least five printing statements that can output some representative lexical errors.

For output, print out the following:

- Print the input program given on the next page exactly as you stored in your input file.
- For each token, if it is a legal one then print the token, its type, and the line number. If illegal, then print the token, an appropriate error message, and the line number.
- Print the content of SYMTAB.

Your program must scan the whole source program, finding all legal and illegal tokens. Run your program on the following input:

```
package a;
package b;
# packages defined
import a.xyz; import b.c…67; # imports
abstract class {
val a, b, c : real;
def 1x (y, w) { y <= w; };
while (not ( true or false)) return (47 * (x + 25));
}
protected object {
val i, j, k : int;
if (@ x 25.2.5) case i = j + k * 5 => print (i);
else in (i, j, k);
} # end object
private class {
val tt, ff: bool;
return (not (true or @ x 5) and false);
}
$
```

The following will be considered for grading purpose:

- Correctness as to whether your program has been designed as instructed above and whether your program runs as intended.
- Documentation, as is usual in any software design, and neatness.

Submit your program and output on or before the due date. Penalty for late submission: 20% per calendar day. A partial credit of no more than half the full credit can be given for an incomplete work.

This project, as well as all other student activities in this course, is an individual (not group) assignment. Plagiarism will result in academic misconduct charges.

Assignment handed out on February 8, 2022.