Feb-08-2017

Regular expression in Java

Purpose

Get familiar with regular expression. Understand the wide application of regular expression.

Assignment specification

Your job is to count the number of identifiers in programs written in our Tiny language.
You should pick out the identifiers from a text file, and write the output to a text file (named A1.output). Note that the output file should contain a line like "identifiers:5" . Here are the sample input and output files.The input will have multiple lines. Please note that in this sample program the following are not counted as identifiers:

> A41, input, output: they are quoted hence they are not treated as identifiers;
> INT, READ etc.: They are keywords used in our Tiny language hence they should not be picked up.

Here are the test cases for the assignment: case 1, case 2, case 3, case 4, case 5, case 6. (ID counts: 5 4 6 7 8 9)
In this assignment you can suppose that there are no comments in the programs.

In the output file you should only write "identifiers:" followed by the number of identifiers. If there are multiple occurrences of an identifier in the input, you should only count it once. Don't write anything else into the output file.

You will write two different programs to do this:

1. Program A11.java is not supposed to use regular expressions, not regex package, not the methods involvoing regular expression in String class or other classes. Hence it will rely on StringTokenizer. Please refer to API JavaDoc for more details of the StringTokenizer specification.
2. Program A12.java will use java.util.regex. Two useful links to start with are JavaDoc of regex and a tutorial for Java regex.
   In A12, you should not use StreamTokenizer or StringTokenizer.

Your programs should be able to run by typing:

```
%javac A11.java
%java A11  A1.tiny
%javac A12.java
%java A12 A1.tiny
```

In this assignment, the output should be in a file called "A1.output". You should not use keyboard input. The input file name will be provided as the argument of the program, while the output file name is hard coded in your programs. i.e., your code regarding input and output can be like the following:

```
          ...  new BufferedReader(new FileReader(args[0]));
          ...  new BufferedWriter(new FileWriter("A1.output"));
```