1X1: PROGRAMMING LANGUAGES

Assignment 1

Due Friday, January 24 in the Assignment 1 Dropbox on E3 at 11:45 pm

No Late Assignments will be accepted.

OVERVIEW

As mentioned in lecture, all of the languages we will study this quarter are interpreted.

An interpreter translates programs written in one language into computational actions, effectively running the program being interpreted. Code Analysis is the first phase in Interpretation, which ensures the source program is syntactically & semantically correct.

This assignment requires you to implement the syntactic analysis phase of Interpretation; you will implement the interpreter's computational engine in a future assignment.

SIMPLESEM INSTRUCTION FORMAT

The interpreter is being written for a language named SIMPLESEM. The SIMPLESEM instruction set contains four op-codes: `set`, `jump`, `jumpt`, and `halt`. The instruction formats are as follows (a future assignment specification will explain each instruction's semantics):

```
set destination, source

jump destination

jumpt destination, boolean-condition

halt
```

To perform syntactic analysis of SIMPLESEM you will:

- Open & Read a SIMPLESEM program
- Use Java to identify & tokenize keywords, symbols, and expressions
- Use the SIMPLESEM grammar below to parse a program according to language rules

SIMPLESEM GRAMMAR

The grammar for SIMPLESEM is as follows:

<Program>→ <Statement> {<Statement>}

<Statement> → <Set> | <Jump> | <Jumpt> | **`halt`**

<Set>→ **`set`** `(write` | <Expr>)`,` (`read` |<Expr>)

<Jump>→ **jump** <Expr>

<Jumpt>→ **jumpt** <Expr> **,** <Expr> (**!=** | **==** | **>** | **<** | **>=** | **<=**) <Expr>

<Expr>→ <Term> {(**+** | **-**) <Term>}

<Term>→ <Factor> {(***** | **/** | **%**) <Factor>}

<Factor>→ <Number> | **D [** <Expr> **]** | **(** <Expr> **)**

<Number>→ **0**| (1..9){0..9}

Guide to EBNF: |— separate alternate choices, ()—choose 1, {}—choose 0 or more, **keyword/symbol**—what is in bold **CourierNew** font.

SYNTACTIC ANALYSIS

To ensure the file is syntactically correct according to the SIMPLESEM grammar, you will need to complete the following tasks :

- Open input file for reading (NAME: provided as command-line argument; FORMAT: one SIMPLESEM statement per line)
- As described in lecture & discussion, use recursion and looping to correctly parse SIMPLESEM grammar rules.

Note: To syntactically analyze the file, you may use Java's Regular Expressions utility, but this is not necessary. More information on Java's regular expressions can be found here: http://java.sun.com/developer/technicalArticles/releases/1.4regex/

OUTPUT

Using the provided `void printRule(String)` function, print the name of each non-terminal entered. Please see the sample `.out` for example output.

SUBMISSION

Submit the following items to the Assignment #1 Dropbox on E3:

- A single zip file labeled with your Student ID, containing ONLY .java files needed to implement this project.
- All projects **must** be able to execute via command line using the following format:

```
java SIMPLESEM Program#.S
```