Interpreter Assignment #2: Statements

Issued: Tuesday, September 30
Due: Tuesday, October 28

Purpose

This assignment asks you to extend your Interpreter Assignment #1 interpreter.

Grammar

As before, your interpreter employs an ad-hoc scanner and a recursive-descent parser. The parser builds a strongly typed parse tree, which is then traversed and evaluated. A grammar for the extended source language is:

```
1   prog    : block
2   block   : stmt ';' block
3           | stmt
4   stmt    : assn
5           | 'rd' id
6           | 'wr' expr
7           | 'if' boolexpr 'then' stmt
8           | 'if' boolexpr 'then' stmt 'else' stmt
9           | 'while' boolexpr 'do' stmt
10          | 'begin' block 'end'
11  assn    : id '=' expr
12  expr    : term addop expr
13          | term
14  term    : fact mulop term
15          | fact
16  fact    : id
17          | num
18          | '(' expr ')'
19          | '-' fact
20  boolexpr : expr relop expr
21  addop   : '+'
22          | '-'
```

```
23  mulop    : '*'
24           | '/'
25  relop    : '<'
26           | '<='
27           | '>'
28           | '>='
29           | '<>'
30           | '=='
```

Assignment

There are several parts:

- Extend your scanner to recognize the new keywords and operators.

- Extend your parser to recognize the new statements and expressions.

- Extend your evaluator to execute the new constructs.

    - You can represent boolean values as double values (e.g., 1.0 and 0.0);
    - For I/O, read from `System.in` (hint: use a `Scanner`) and write to `System.out`.

- Test your solution thoroughly. Add tests to your test suite. The quality of your suite will influence your grade.