

## Interpreter Assignment #2: Statements

**Issued:** Tuesday, March 2

**Due:** Thursday, March 18

### 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:

```

prog      : block
block     : stmt ';' block
          | stmt
stmt      : assn
          | 'rd' id
          | 'wr' expr
          | 'if' boolexpr 'then' stmt
          | 'if' boolexpr 'then' stmt 'else' stmt
          | 'while' boolexpr 'do' stmt
          | 'begin' block 'end'
assn      : id '=' expr
expr      : term addop expr
          | term
term      : fact mulop term
          | fact
fact      : id
          | num
          | '(' expr ')'
```

```

        | '-' fact
boolexpr : expr relop expr
addop    : '+'
        | '-'
mulop     : '*'
        | '/'
relop     : '<'
        | '<='
        | '>'
        | '>='
        | '<>'
        | '=='

```

## 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 `JDK 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.