**UNIVERSITY OF CAMBRIDGE COMPUTER LABORATORY**

**Computer Science Tripos Part II (General) 2007–2008**

# Assessed Exercise No. 1

## Syntax Analysis and Evaluation of Simple Integer Expressions

### Problem Outline

A Java program is required to analyse and evaluate expressions such as:

`50 - (6*9)/3`

### General Syntax

The program should accept data with the following syntax:

```
     <the whole thing> ::= <expression sequence> <terminator>
<expression sequence> ::= <expression> | <expression> <separator> <expression sequence>
          <expression> ::= <simple expression> | <empty>
   <simple expression> ::= <item> | <item> <operator> <simple expression>
                <item> ::= <number> | (<simple expression>)
              <number> ::= <digit> | <digit> <number>
               <digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
            <operator> ::= + | - | * | /
          <terminator> ::= End of Data (i.e. the Java EOFException exception)
           <separator> ::= ;
               <empty> ::=
```

A space or newline is allowed anywhere except between adjacent digits in a number.

### Semantics

The operators are as in Java `int` arithmetic so division using `/` involves truncation towards zero. The syntax implies that, in the absence of brackets, evaluation of an individual expression is right to left; thus `8-3+1` has the value 4 and not 6. There are no monadic operators.

### Suggested Strategy

It is recommended that the program should include:

1. A lexical analysis method which takes in one character at a time and yields a sequence of symbols.
2. Syntax tree building methods which take the sequence of symbols and generate a tree.
3. An evaluation method which processes the tree and yields a value.

**Requirements**

Output from typical data should be roughly as follows:

```
Expression 1:  value is 287
Expression 2:  value is -74
Expression 3:  unknown character '#' detected.  Abandoned.
Expression 4:  value is 83
Expression 5:  mismatched brackets.  Abandoned.
```

It is perfectly in order to abandon an expression as soon as a syntax error is detected but the program must recover from such an error and proceed to the next expression.

Recovery from errors during evaluation (e.g. division by zero) is desirable but will not insisted upon.

**Documentation and Evidence of Success**

Three items must be handed in:

1. A report
2. A copy of the program
3. Sample data and corresponding output from a run of the program

The report must:

1. be computer printed.
2. document clearly how the program works. This document should include diagrams which describe the principal data structures. The diagrams may be produced by a computer-drawing package or they may be drawn neatly by hand.

The report will probably be 1000–2000 words long. Much less than 1000 words is unlikely to give sufficient explanation. Much more than 2000 words is a sign that too much time has been spent on the exercise!

**Deadline and Oral**

Report, program and output from completed exercises must be handed to the Student Administration Secretary by noon on 26 October 2007.

After this deadline, notifications of interview times with an assessor will be issued. The assessor may well ask for the program to be run and may wish to supply challenging test data to exercise the program's ability to deal with special cases.

F.H. King

26 July 2007