CSE 461: Programming Languages Concepts

Prof. G. Tan
Spring 2018

Homework 1: **Due on Jan 19th before class (12:20pm) in Canvas.**
Total: 22 points

Submission: Please submit your homework via Canvas. It's okay if you submit a scanned version of your on-paper answers, but please make sure your scanned version is legible.

1. (5 points) We have the following grammar with the start symbol `<e>`:

   ```
   <e>  ->  <d> | <e> * <e> | <e> / <e>
   <d>  ->  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
   ```

   (a) Show a leftmost derivation for the expression "2 * 3 * 6".

   (b) Show a rightmost derivation for the above expression.

   (c) Show two different parse trees for the above expression.

   (d) The grammar is ambiguous. Show a new grammar that removes the ambiguity and makes "*" and "/" left-associative. Show the parse tree for "2 * 3 / 6" in your new grammar. Argue why this is the only parse tree in the new grammar.

   (e) Show a new grammar that removes the ambiguity and makes "*" and "/" right-associative. Show the parse tree for "2 * 3 / 6" in the new grammar.

2. (3 points) Consider the language consisting of strings that represent the list of numbers separated by commas. For instance, the string "`10,7`" and "`1, 7, 5, 13`" are in the language; also included in the language are lists of a single number (e.g., "`12`"). Write an unambiguous BNF grammar for the language. Briefly explain why your grammar is unambiguous.

3. (3 points). The following grammar for arithmetic expressions allow addition, subtraction, as well as a unary operator "~" for negation; that is, "~8" is interpreted as number negative eight.

```
<e>  ->  <d> | <e> + <e> | <e> - <e> | ~<e>
<d>  ->  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

The grammar is clearly ambiguous. Change the grammar so that "+" and "-" are left-associative and the precedence of "~" is higher than "+" and "-".

4. (4 points) A simplified email address has (i) an account name starting with a letter and continuing with any number of letters or digits; (ii) an @ character; (iii) a host with two or more sequences of letters or digits separated by periods; the last sequence must be a toplevel domain—either 'edu', 'org', or 'com'. Define a context-free grammar to model this language.

5. (4 points) The following grammar discussed in class is for constructing numbers:

```
<n> -> <d> | <n> <d>
<d>  ->  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

Numbers with leading zeros such as 007 belong to the above grammar. Change the grammar so that numbers with leading zeros cannot be derived from the new grammar; however, number 0 itself should still be allowed. As a sanity check, make sure numbers such as 70 and 107 belong to your grammar, while numbers such as 070 do not.

6. (3 points) The following E-BNF grammar is used to specify decimal numbers such as 7.9 and -10.78. Translate it into an equivalent BNF grammar.

```
<expr> -> [-] <int> [.<int>]
<int> -> <digit> {<digit>}
<digit> -> 0 | 1 | ... | 9
```