CSE 461: Programming Languages Concepts

Prof. G. Tan, Spring 2020

Homework 1: **Due on Jan 25th at 6pm in Gradescope.** Total: 22 points.

Submission: Please submit your homework via Gradescope. You can watch a video about how to submit homework via Gradescope below:
https://www.youtube.com/watch?time_continue=1&v=KMPoby5g_nE&feature=emb_logo
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 "7 + 4 - 5"; show every step.

    (b) Show a rightmost derivation for the above expression; show every step.

    (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 "7 + 4 - 5" 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 "7 + 4 - 5" in the new grammar.

2. (3 points) Show the following BNF grammar (with start symbol `<S>`) is ambiguous by giving an example input and drawing its two different parse trees. Give an equivalent unambiguous grammar.

    ```
    <S>  ->  <A>
    <A>  -> <A> and <A> | <id>
    <id> -> a | b | c
    ```

3. (2 points) Consider the language consisting of strings that have n copies of the letter a followed by 2n copies of the letter b where n > 0. For example, the strings abb, aabbbb, and aaabbbbbb are in the language but a, ab, ba and aabbb are not. Give an unambiguous BNF grammar for the language.

4. (4 points) Consider the grammar given bellow:

```
<assign> -> <id> = <expr>
<id>     -> x | y | z
<expr>   -> <expr> + <term> | <term>
<term>   -> <term> * <factor> | <factor>
<factor> -> (<expr>) | <id>
```

   Give a complete grammar that extends the above grammar to include a binary exponentiation operator ** (i.e., b ** n is used in some languages to mean b raised to the n-th power). In this grammar, make the ** operator right-associative and give it a higher precedence over +, but a lower precedence over *. For example, “x + x ** y ** z” should be parsed the same as “x + (x ** (y ** z))”.

5. (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.

6. (4 points) The following E-BNF is the grammar for a simplified version of LISP. Convert it to a BNF grammar. Note in the following “{”, “}”, “[”, “]”, and “|” are meta-symbols of E-BNF, while “(”, “)”, and “.” are terminals.

```
<s-exp> -> <atomic-sym> | ( <s-exp> . <s-exp> ) | ( <s-exp-list> )
<s-exp-list> -> { <s-exp> }
<atomic-sym> -> <letter> { <letter> | <number> }
<letter> -> a | b | ... | z
<number> -> 0 | 1 | ... | 9
```