CMPSC 461: Programming Languages Concepts

Prof. G. Tan, Spring 2022

Homework 1: **Due on Jan 21st at 6pm in Canvas.** Total: 21 points.

Submission: If you submit a scanned version of your on-paper answers, 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 "9 / 3 * 3"; 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 "9 / 3 * 3" 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 "9 / 3 * 3" in the new grammar.

2. (3 points) Is the following BNF grammar ambiguous? Assume the start symbol is `<n>`. If it is ambiguous, give an example input and draw different parse trees for it; also, formulate an unambiguous grammar that is equivalent.

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

3. (2 points) Suppose a language contains strings that have n copies of left parentheses followed by 2n copies of right parentheses, where $n > 0$. For instance, "())" and "(()))" are in the language. Formulate an unambiguous BNF grammar for this language.

4. (3 points) For the simple arithmetic language we studied in class, imagine we add a unary operator "~" for negation; that is, "~9" is interpreted as number negative night. The grammar is then

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

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

5. (4 points) Consider a language for simplified email addresses. The address has three parts. The first part is an account name starting with a letter and continuing with any number of letters or digits. The second part is an @ character. The third part is a host with two or more sequences of letters or digits separated by periods; the last sequence must be a top-level domain—either 'com', 'org', 'gov', or 'net'. Define a context-free grammar to model this language.

6. (4 points) The following grammar defines the scientific notation for non-negative floating point numbers. It is in the E-BNF notation and "[", "]", "(", ")", "|", "{", and "}" are meta-symbols of E-BNF. Convert this E-BNF grammar to a BNF grammar.

```
<SNFloat> -> <Float> [E<Exponent>]
<Float> -> <NonZeroDigit> [.<Num>]
<Exponent> -> [(+|-)]<Num>
<Num> -> <Digit>{<Digit>}
<Digit> -> 0 | <NonZeroDigit>
<NonZeroDigit> -> 1 | 2 | ... | 9
```