Homework: Context-Free Grammar

Due-date: Jan 26 at 11:59pm Submit online on Blackboard LS (**Format: pdf**; No other format allowed)

Homework must be individual's original work. Collaborations and discussions of any form with any students or other faculty members are not allowed. If you have any questions and/or concerns, post them on Piazza and/or ask our TA or me.

Learning Outcomes

- Knowledge and application of Grammars
- Knowledge of Mathematical Formalisms in Programming Languages

Questions

1. True/False. Justify your answers.

   (a) Ambiguous grammar may not necessarily lead to ambiguity in meaning of the sentences generated by the grammar.

   (b) Ambiguity in the meaning of a grammatically correct sentence implies that the grammar, from which the sentence is generated, is ambiguous.

   (c) For an unambiguous grammar, left-most derivation and right-most derivation strategies must result in the same parse tree.

   (d) The result of the Java statement: $x = ++y * 2$; is "x is assigned to 8 when y is equal to 3".

   (8)

2. Consider the following grammar with

   - terminals: $\{x, y, z, +, -, *, /\ \}$
   - non-terminals: $\{E, T, F, V\}$
   - start symbol: E
   - production rules:

$$\begin{array}{lcl} E & \rightarrow & E\ +\ T \mid E\ -\ T \mid T \\ T & \rightarrow & T\ *\ F \mid T\ /\ F \mid F \\ F & \rightarrow & V \\ V & \rightarrow & x \mid y \mid z \end{array}$$

   (a) What are the associativities of "+" and "-" operators? Explain your answer.

   (b) If a grammatically correct (according to the above grammar) expression has more "+" operations than "-" operations, then as per the above grammar, can we infer that the semantics of the "+" operations will be evaluated before that of "-" operations? Explain your answer.

   (c) Is $x + y/z * x + y - x$ grammatically correct as per the above grammar? Explain your answer.

   (7)

3. Consider the following grammar for a program

```
Prog        → StmtSeq
StmtSeq     → Stmt; StmtSeq | Stmt;
Stmt        → AssignStmt | WhileStmt
AssignStmt  → V = E
WhileStmt   → while CExpr do StmtSeq
CExpr       → BExpr | CExpr && CExpr | ! CExpr
BExpr       → E > E | E < E | E == E
```

In the above,

- production rules for E and V are obtained from the previous question
- terminals: $\{x, y, z, +, -, *, /\,;\,, \texttt{while}, \texttt{do}, >, <, =, ==, \&\&, !\}$
- non-terminals: Any symbol that appears on the left-hand side of the production rules
- start symbol: `Prog`

(a) How many programs can be generated by the above grammar?

(b) Is the above grammar ambiguous? Explain your answer (if your answer is "no", then provide arguments to support to your answer; if your answer is "yes", present a program which has at least two distinct parse trees because of which the semantics of the program may become ambiguous as well).

(5)

4. Consider a language of logical expressions.

(a) A logical expression is either a quantified expression or a boolean expression or a variable expression or a propositional constant

(b) A variable expression is either x or y or z.

(c) A propositional constant is either $true$ or $false$.

(d) If A is a logical expression then $\exists\ v.A$ and $\forall\ v.A$ are quantified expressions such that $v \in \{x, y, z\}$

(e) If A is a logical expression and B is a logical expression then $\neg\ A$, $A\ \wedge\ B$ and $A\ \Rightarrow\ B$ are boolean expressions

The language has the following operator precedence order (in descending order of precedence):

(a) $\neg$

(b) $\wedge$

(c) $\Rightarrow$

The language also has operator associativity: $\wedge$ is left-associative and $\Rightarrow$ is right-associative.

Write unambiguous production rules for the grammar that generates the strings in the above language.

(10)