**CS 421 — LL Grammar Problems**

**Question 1:** What advantage does it give the programmer if a grammar turns out to be LL?

**Question 2:** What are the two conditions that will cause a grammar to not be LL?

**Question 3:** Consider the following grammar.

\[
S \rightarrow x E a \\
| y E z \\
E \rightarrow x F \\
| x E q \\
F \rightarrow q \\
| F z
\]

This grammar is not LL. There are two reasons for that. What are they?

**Question 4:** Convert the following grammar into an equivalent LL grammar.

\[
S \rightarrow x E a \\
| y E z \\
E \rightarrow x F \\
| x E q \\
F \rightarrow q \\
| F z
\]

**Question 5:** Convert the following non-LL grammar into an equivalent LL grammar.

\[
S \rightarrow S x \\
| a E \\
E \rightarrow y a y \\
| y a z
\]

**Question 6:** Convert the following non-LL grammar into an equivalent LL grammar.

\[
S \rightarrow y E \\
| y z \\
E \rightarrow E z z \\
| x a a
\]

**Question 7:** Convert the following non-LL grammar into an equivalent LL grammar.

\[
S \rightarrow S x \\
| a E \\
E \rightarrow x y \\
| x z
\]

**Question 8:** Convert the following non-LL grammar into an equivalent LL grammar.

\[
S \rightarrow S x \\
| x E \\
E \rightarrow y y \\
| y z
\]

**Question 9:** Convert the following non-LL grammar into an equivalent LL grammar.

\[
S \rightarrow y E \\
| y z \\
E \rightarrow E z z \\
| x a a
\]

**Question 10:** Convert the following non-LL grammar into an equivalent LL grammar.

\[
S \rightarrow S x \\
| a E \\
E \rightarrow z b b \\
| z b z
\]
1 Solutions to exercises

Solution 1 If the grammar is LL, then we can write a parser for it very simply using recursive descent.

Solution 2
1. two rules for the same symbol that have overlapping first sets (The “common prefix” problem.)
2. a left recursive rule

Solution 3 The $E$ productions share a common prefix, $x$; and one of the $F$ rules is left recursive.

Solution 4

Solution 5

Solution 6

Solution 7

Solution 8

Solution 9

Solution 10