The syntax of programming languages is usually given as a context-free grammar. To keep things simple we consider a very simple language which we call P. The syntax of P is given by the following CFG $G = (V, \Sigma, S, P)$:

- $V = \{ \text{Prog}, \text{Stmt}, \text{Stmts}, \text{Expr}, \text{OpName}, \text{Num} \}$
- $\Sigma = \{\{,\},,(,),+,\ast,\div,\text{if},\text{while},\text{print},=,;,0,1,x,y,z\}$
- $S = \text{Prog}$
- $P$ is given by:

\[
\begin{align*}
\text{Prog} & \rightarrow \{\text{Stmts}\} \\
\text{Stmts} & \rightarrow \epsilon | \text{Stmt} \text{Stmts} \\
\text{Stmt} & \rightarrow \text{Name} = \text{Expr}; \\
& \quad | \text{if} (\text{Expr}) \text{Stmt} \\
& \quad | \text{while} (\text{Expr}) \text{Stmt} \\
& \quad | \text{print} \text{Expr} ; \\
& \quad | ; \\
& \quad | \text{Prog} \\
\text{Expr} & \rightarrow \text{Name} | \text{Num} | (\text{Expr} \text{Op} \text{Expr}) \\
\text{Name} & \rightarrow x | y | z \\
\text{Op} & \rightarrow + | * | - | \div \\
\text{Num} & \rightarrow 0 | 1
\end{align*}
\]

Note that we have only a finite set of names and numeric constants. In a real language the syntax of those would have been specified using a regular language (i.e. a 2-level approach would have been used).

1. Decide which of the following are correct P programs and give a reason for programs you reject.

(a) \{ print x ; \{ print y ; \} \}
(b) \{ if( x ) ; ; \}
(c) \{ x = x + 1 ; print x ; \}
(d) \{ print y \}
(e) \{ while ( z ) \{ x = ( x + 1 ) ; \} print y ; \}
(f) \{ \{ \} \}
(g) \{ \text{print} (0); \}
(h) \{ \text{if} (x = 0) \text{print} x; \}
(i) \text{print} x;
(j) \{ \text{while} (x) \{ \} \text{while} (y) \{ \} \}

2. Give derivations and parse trees for the following P programs:

(a) \{ \text{while} (x) \text{print} y; \text{print} z; \}
(b) \{ \text{while} (x) \{ \text{print} y; \text{print} z; \} \}