CS 5353 Compiler Construction
Spring 2015 Language Syntax - LL(1) Format

$$
\begin{array}{rcll}
\langle program \rangle & ::= & \langle func \rangle \langle func\_list \rangle & (1) \\
\langle func\_list \rangle & ::= & \langle func \rangle \langle func\_list \rangle & (2) \\
 & ::= & \varepsilon & (3) \\
\langle func \rangle & ::= & \ll id \gg (\langle vars \rangle) \text{ -> } < \langle return\_list \rangle > & \\
 & & \{ \langle var\_decs \rangle \langle block \rangle \} & (4) \\
 & ::= & forward \ll id \gg (\langle vars \rangle) \text{ -> } < \langle return\_list \rangle > & (5) \\
\langle vars \rangle & ::= & \langle var\_dec \rangle \langle vars\_tail \rangle & (6) \\
 & ::= & \varepsilon & (7) \\
\langle var\_dec \rangle & ::= & \langle type \rangle \ll var\_id \gg \langle array\_dec \rangle & (8) \\
\langle vars\_tail \rangle & ::= & , \langle var\_dec \rangle \langle vars\_tail \rangle & (9) \\
 & ::= & \varepsilon & (10) \\
\langle var\_decs \rangle & ::= & \langle vars \rangle ; & (11) \\
 & ::= & \varepsilon & (12) \\
\langle array\_dec \rangle & ::= & [\ll int \gg] & (13) \\
 & ::= & \varepsilon & (14) \\
\langle return\_list \rangle & ::= & \langle type \rangle \langle return\_tail \rangle & (15) \\
 & ::= & \varepsilon & (16) \\
\langle return\_tail \rangle & ::= & , \langle type \rangle \langle return\_tail \rangle & (17) \\
 & ::= & \varepsilon & (18) \\
\langle type \rangle & ::= & int & (19) \\
 & ::= & char & (20) \\
 & ::= & bool & (21) \\
\langle block \rangle & ::= & \langle stmt \rangle \langle block \rangle & (22) \\
 & ::= & \varepsilon & (23)
\end{array}
$$

$$
\begin{aligned}
\langle stmt \rangle &::= \langle var\_list \rangle := \langle arg\_list \rangle; && (24)\\
&::= if(\langle expr \rangle)\{\langle block \rangle\}\langle else\_tail \rangle && (25)\\
&::= while(\langle expr \rangle)\{\langle block \rangle\} && (26)\\
&::= write(\langle write\_list \rangle); && (27)\\
&::= read(\langle var\_list \rangle); && (28)\\
&::= return\langle arg\_list \rangle; && (29)\\
&::= \langle function\_call \rangle; && (30)\\
\langle var\_list \rangle &::= \langle var\_ref \rangle \langle var\_tail \rangle && (31)\\
\langle var\_tail \rangle &::= , \langle var\_list \rangle && (32)\\
&::= \varepsilon && (33)\\
\langle var\_ref \rangle &::= \ll var\_id \gg \langle array\_ref \rangle && (34)\\
\langle array\_ref \rangle &::= [\langle expr \rangle] && (35)\\
&::= \varepsilon && (36)\\
\langle else\_tail \rangle &::= else\{\langle block \rangle\} && (37)\\
&::= \varepsilon && (38)\\
\langle arg\_list \rangle &::= \langle arg \rangle \langle arg\_tail \rangle && (39)\\
\langle arg \rangle &::= \ll char \gg && (40)\\
&::= \langle expr \rangle && (41)\\
\langle arg\_tail \rangle &::= , \langle arg\_list \rangle && (42)\\
&::= \varepsilon && (43)\\
\langle write\_list \rangle &::= \langle write\_item \rangle \langle write\_tail \rangle && (44)\\
\langle write\_item \rangle &::= \ll char \gg && (45)\\
&::= \ll string \gg && (46)\\
&::= \langle expr \rangle && (47)\\
\langle write\_tail \rangle &::= , \langle write\_list \rangle && (48)\\
&::= \varepsilon && (49)\\
\langle expr \rangle &::= \langle b\_expr \rangle \langle expr\_1 \rangle && (50)\\
\langle expr\_1 \rangle &::= | \ \langle b\_expr \rangle \langle expr\_1 \rangle && (51)\\
&::= \& \ \langle b\_expr \rangle \langle expr\_1 \rangle && (52)\\
&::= \varepsilon && (53)
\end{aligned}
$$

$$
\begin{aligned}
\langle b\_expr \rangle &::= \langle n\_expr \rangle \langle b\_expr\_1 \rangle && (54)\\
\langle b\_expr\_1 \rangle &::= < \langle n\_expr \rangle && (55)\\
&::= > \langle n\_expr \rangle && (56)\\
&::= <= \langle n\_expr \rangle && (57)\\
&::= >= \langle n\_expr \rangle && (58)\\
&::= = \langle n\_expr \rangle && (59)\\
&::= \ != \langle n\_expr \rangle && (60)\\
&::= \varepsilon && (61)\\
\langle n\_expr \rangle &::= \langle term \rangle \langle n\_expr\_1 \rangle && (62)\\
\langle n\_expr\_1 \rangle &::= + \langle term \rangle \langle n\_expr\_1 \rangle && (63)\\
&::= - \langle term \rangle \langle n\_expr\_1 \rangle && (64)\\
&::= \varepsilon && (65)\\
\langle term \rangle &::= \langle factor \rangle \langle term\_1 \rangle && (66)\\
\langle term\_1 \rangle &::= * \langle factor \rangle \langle term\_1 \rangle && (67)\\
&::= / \langle factor \rangle \langle term\_1 \rangle && (68)\\
&::= \varepsilon && (69)\\
\langle factor \rangle &::= \langle sub\_factor \rangle \langle factor\_1 \rangle && (70)\\
\langle factor\_1 \rangle &::= \% \langle sub\_factor \rangle \langle factor\_1 \rangle && (71)\\
&::= \varepsilon && (72)\\
\langle sub\_factor \rangle &::= \langle base \rangle \langle sub\_factor\_1 \rangle && (73)\\
\langle sub\_factor\_1 \rangle &::= \hat{} \ \langle sub\_factor \rangle && (74)\\
&::= \varepsilon && (75)\\
\langle base \rangle &::= \langle var\_ref \rangle && (76)\\
&::= \ll int \gg && (77)\\
&::= \ll bool \gg && (78)\\
&::= !\langle base \rangle && (79)\\
&::= (\langle expr \rangle) && (80)\\
&::= \langle function\_call \rangle && (81)\\
\langle function\_call \rangle &::= \ll id \gg (\langle call\_list \rangle) && (82)\\
\langle call\_list \rangle &::= \langle arg\_list \rangle && (83)\\
&::= \varepsilon && (84)
\end{aligned}
$$