Solutions to Written Assignment 2

1. CFGs for languages.

   (a) All nonempty strings that start and end with the same symbol.

$$\begin{aligned} S &\rightarrow 0 \mid 1 \mid 0A0 \mid 1A1 \\ A &\rightarrow A0 \mid A1 \mid \varepsilon \end{aligned}$$

   (b) All strings that contain more 1s than 0s.

$$\begin{aligned} S &\rightarrow A1 \mid MS \mid SMA \\ A &\rightarrow A1 \mid \varepsilon \\ M &\rightarrow \varepsilon \mid MM \mid 0M1 \mid 1M0 \end{aligned}$$

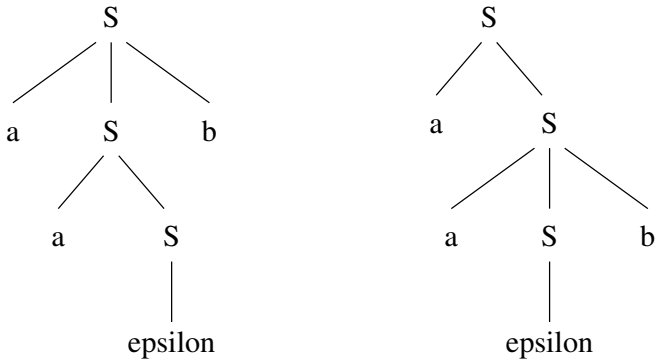
   (c) All palindromes.

$$S \rightarrow \varepsilon \mid 0 \mid 1 \mid 0S0 \mid 1S1$$

2. Ambiguous grammars.

   (a) The strings in this language consist of a sequence of n a's followed by m b's, for any n and m such that $n \geq m \geq 0$.

   (b) The string aab can be parsed in two ways:



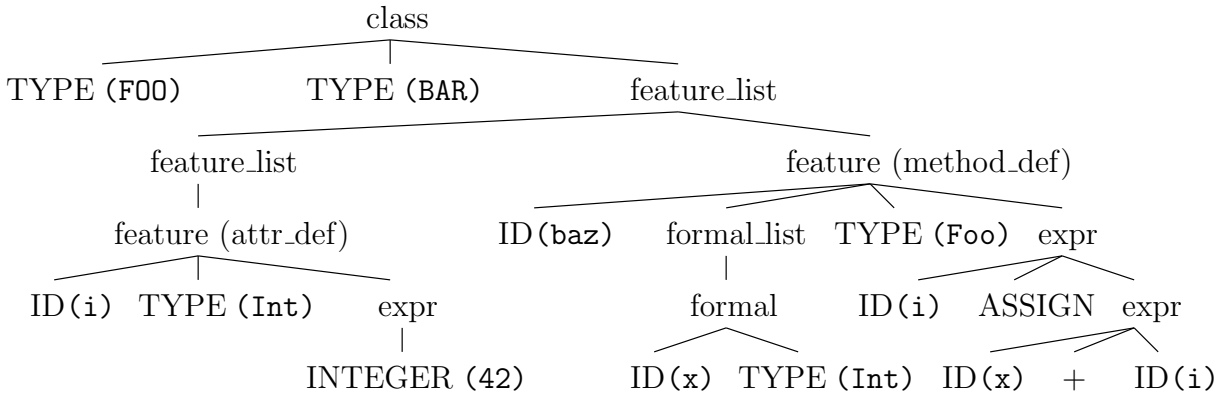
   (c)

$$\begin{aligned} S &\rightarrow aSb \\ S &\rightarrow T \\ T &\rightarrow aT \\ T &\rightarrow \epsilon \end{aligned}$$

3. Parse trees.

   The figure below shows a sample parse tree for this class definition. In the Cool Reference Manual,

the specification of the syntax for Cool is not given as a pure CFG, and so some productions have been introduced here to capture the regular expression notation used in the specification. The goal of this question was to practice working with the general structure of a parse tree for a language such as Cool, and as such this sample tree is primarily intended to illustrate the overall structure of the parse tree, as opposed to the details of all the nodes in the tree.



4. Description of language from CFG.

   (a) All binary strings that contain at least three 1s.

   (b) All binary strings that contain an odd number of symbols.

5. $LL(1)$ vs. $LL(2)$ grammars.

$$S \rightarrow ab \mid ac$$

6. $LL(1)$ parsing.

   (a)

$$\begin{aligned} E &\rightarrow \mathbf{id}X \\ X &\rightarrow \varepsilon \mid (A) \mid [E] \\ A &\rightarrow EY \\ Y &\rightarrow \varepsilon \mid \mathbf{;}\ A \end{aligned}$$

   (b) The First and Follow sets of the non-terminals are as follows.

$$\begin{array}{ll} \text{First}(E) = \{\mathbf{id}\} & \text{Follow}(E) = \{\mathbf{\$},], \mathbf{;},)\} \\ \text{First}(X) = \{\varepsilon, (, [\} & \text{Follow}(X) = \{\mathbf{\$},], \mathbf{;},)\} \\ \text{First}(A) = \{\mathbf{id}\} & \text{Follow}(A) = \{)\} \\ \text{First}(Y) = \{\varepsilon, \mathbf{;}\} & \text{Follow}(Y) = \{)\} \end{array}$$

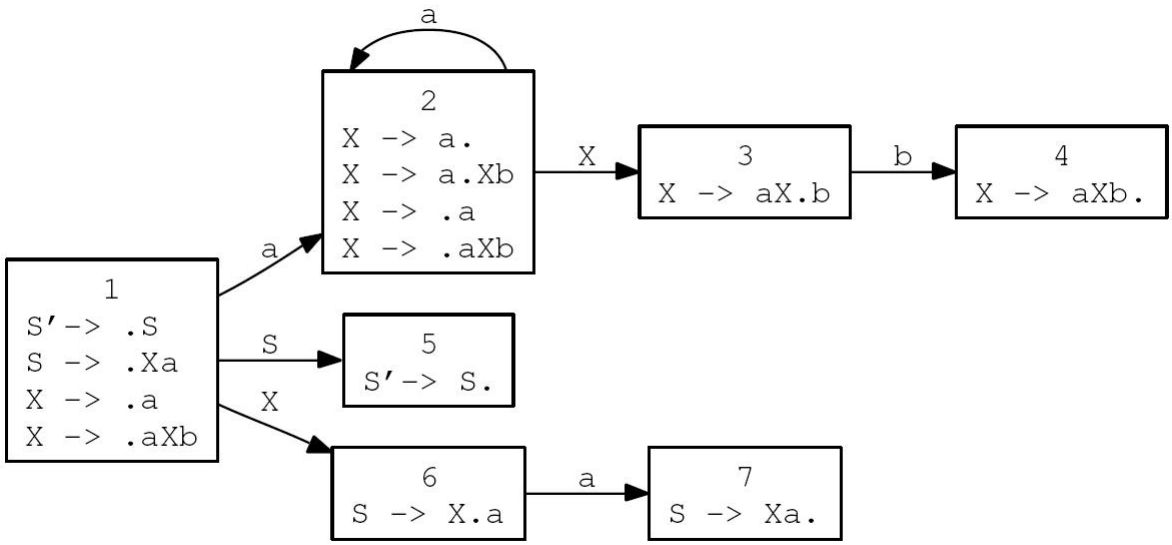
   Here is an LL(1) parsing table for the grammar.

	id	(	)	[	]	;	**$**
E	$E \rightarrow \mathbf{id}X$						
X		$X \rightarrow (A)$	$X \rightarrow \varepsilon$	$X \rightarrow [E]$	$X \rightarrow \varepsilon$	$X \rightarrow \varepsilon$	$X \rightarrow \varepsilon$
A	$A \rightarrow EY$						
Y			$Y \rightarrow \varepsilon$			$Y \rightarrow \mathbf{;}\ A$	

(c)

Stack	Input	Action
E**\$**	**id(id[id]; id)\$**	$E \rightarrow$ **id**X
id$X$**\$**	**id(id[id]; id)\$**	terminal **id**
X**\$**	**(id[id]; id)\$**	$X \rightarrow$ **(**A**)**
(A**)\$**	**(id[id]; id)\$**	terminal **(**
A**)\$**	**id[id]; id)\$**	$A \rightarrow EY$
EY**)\$**	**id[id]; id)\$**	$E \rightarrow$ **id**X
idXY**)\$**	**id[id]; id)\$**	terminal **id**
XY**)\$**	**[id]; id)\$**	$X \rightarrow$ **[**E**]**
[E**]**Y**)\$**	**[id]; id)\$**	terminal **[**
E**]**Y**)\$**	**id]; id)\$**	$E \rightarrow$ **id**X
id$X$**]**Y**)\$**	**id]; id)\$**	terminal **id**
X**]**Y**)\$**	**]; id)\$**	$X \rightarrow \varepsilon$
]Y**)\$**	**]; id)\$**	terminal **]**
Y**)\$**	**; id)\$**	$Y \rightarrow$ **;** A
; A**)\$**	**; id)\$**	terminal **;**
A**)\$**	**id)\$**	$A \rightarrow EY$
EY**)\$**	**id)\$**	$E \rightarrow$ **id**X
idXY**)\$**	**id)\$**	terminal **id**
XY**)\$**	**)\$**	$X \rightarrow \varepsilon$
Y**)\$**	**)\$**	$Y \rightarrow \varepsilon$
)\$	**)\$**	terminal **)**
\$	**\$**	Accept

7. (a) The following figure shows a DFA for viable prefixes of the grammar.



(b) The First and Follow sets of the non-terminals in the grammar are as follows.

$\text{First}(S) = \{\mathbf{a}\}$ $\text{Follow}(S) = \{\mathbf{\$}\}$
$\text{First}(X) = \{\mathbf{a}\}$ $\text{Follow}(X) = \{\mathbf{a}, \mathbf{b}\}$

In the DFA state 2, one valid action for an SLR(1) parser would be to shift **a**. Also, the parser could reduce using the production $X \rightarrow \mathbf{a}$ on the lookahead symbol **a**, because **a** is in the Follow set of the non-terminal X.

(c)

Configuration	DFA Halt State		Action
$\mid$ **a a b a $**	1		Shift **a**
a $\mid$ **a b a $**	2	shift-reduce conflict	Shift **a**
a a $\mid$ **b a $**	2		Reduce $X \rightarrow \mathbf{a}$
a X $\mid$ **b a $**	3		Shift **b**
a X **b** $\mid$ **a $**	4		Reduce $X \rightarrow \mathbf{a}X\mathbf{b}$
X $\mid$ **a $**	6		Shift **a**
X **a** $\mid$ **$**	7		Reduce $S \rightarrow X\mathbf{a}$
S $\mid$ **$**	5		Reduce $S' \rightarrow S$
S' $\mid$ **$**			Accept

(d) The item $X \rightarrow \mathbf{.}$ will appear in the DFA state 2. Now, in state 2, there will be two possible reductions on the lookahead symbol **a**: $X \rightarrow \mathbf{a}$ and $X \rightarrow \varepsilon$.