

Computer Science 414
Spring 2010
Final Practice Problems
Solutions

1. For the following grammar, remove left recursion, left-factor, give the first and follow sets for each non-terminal, and give the parse table.

Terminals = {a, b, c, \$}
 Non-Terminals = { S' , S , A , B , C }
 Rules = (0) $S' \rightarrow S\$$
 = (1) $S \rightarrow AB$
 = (2) $A \rightarrow Aab$
 = (3) $A \rightarrow \epsilon$
 = (4) $B \rightarrow BC$
 = (5) $B \rightarrow b$
 = (6) $C \rightarrow ccA$
 = (7) $C \rightarrow cb$
 Start Symbol = S'

After removing left recursion:

Terminals = {a, b, c, \$}
 Non-Terminals = { S' , S , A , B , C }
 Rules = (0) $S' \rightarrow S\$$
 = (1) $S \rightarrow AB$
 = (2) $A \rightarrow \epsilon$
 = (3) $A \rightarrow abA$
 = (4) $B \rightarrow bB'$
 = (5) $B' \rightarrow CB'$
 = (6) $B' \rightarrow \epsilon$
 = (7) $C \rightarrow cC'$
 = (8) $C' \rightarrow cA$
 = (9) $C' \rightarrow b$
 Start Symbol = S'

Non-Terminal	First	Follow
S'	a, b	
S	a, b	\$
A	a, ϵ	b, c, \$
B	b	\$
B'	c, ϵ	\$
C	c	c, \$
C'	c, b	c, \$

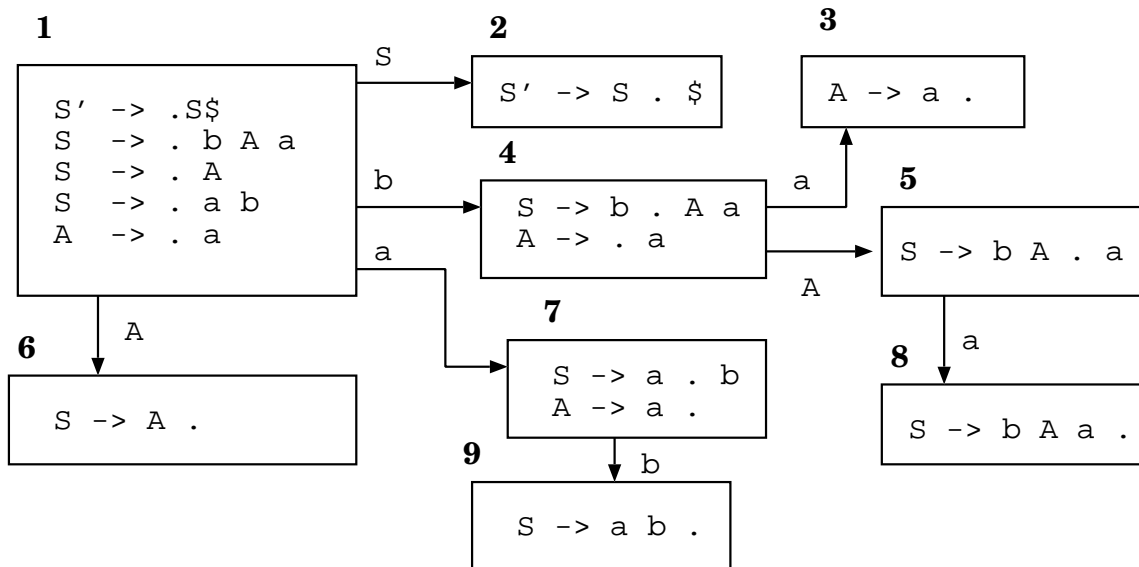
Non-Terminal	a	b	c	\$
S'	$S' \rightarrow S\$$	$S' \rightarrow S\$$		
S	$S \rightarrow ABC$	$S \rightarrow ABC$		
A	$A \rightarrow abA$	$A \rightarrow \epsilon$	$A \rightarrow \epsilon$	$A \rightarrow \epsilon$
B		$B \rightarrow bB'$		
B'			$B' \rightarrow CB'$	$B' \rightarrow \epsilon$
C			$C \rightarrow cC'$	
C'		$C' \rightarrow b$	$C' \rightarrow cA$	

2. Give the LR(0) states and transitions, Follow sets for each non-terminal and the SLR(1) parse table for the following grammars.

(a)

Terminals = {a, b, \$}
 Non-Terminals = {S, S', A}
 Rules = (0) S' → S\$
 = (1) S → bAa
 = (2) S → A
 = (3) S → ab
 = (4) A → a
 Start Symbol = S'

	First	Follow
S'	a,b	
S	a,b	\$
A	a	a, \$

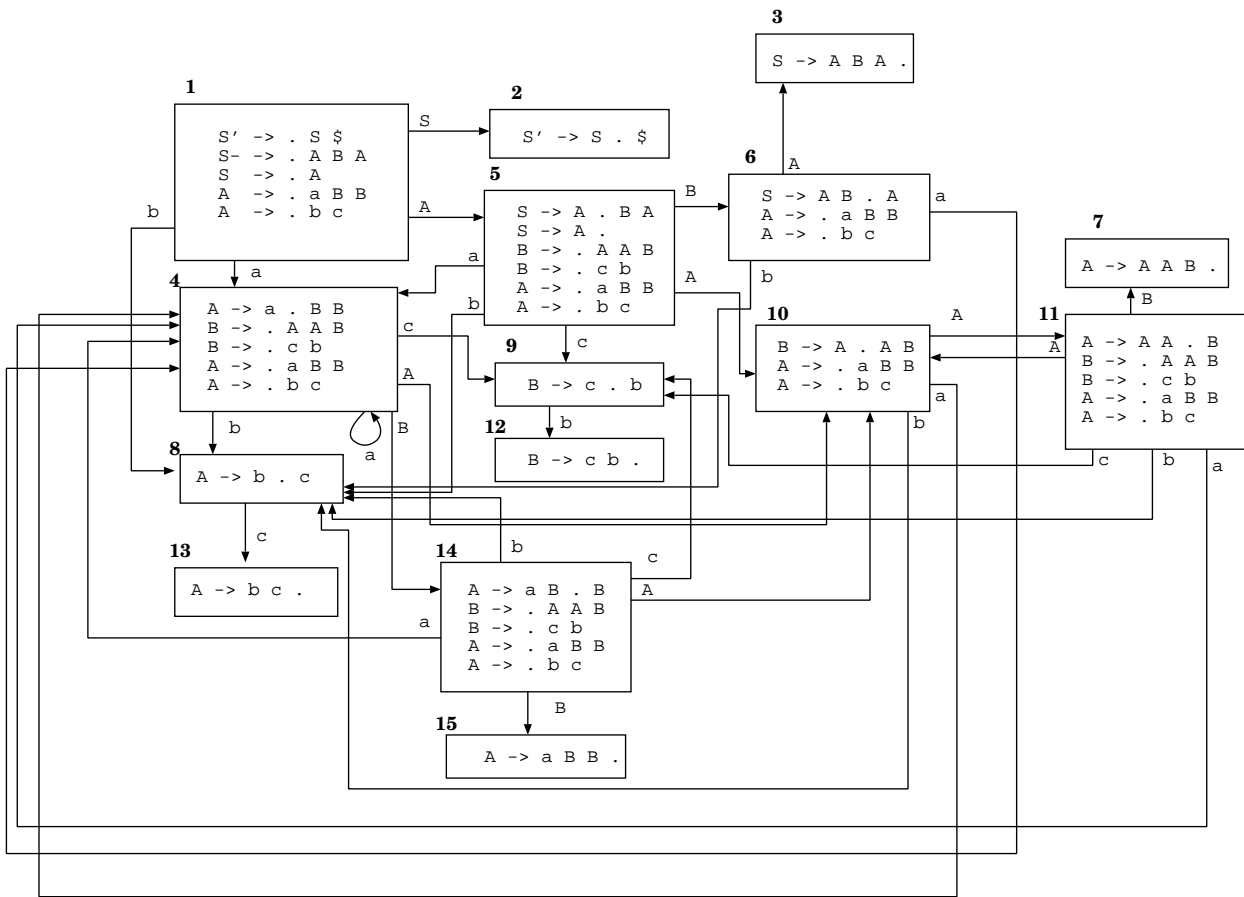


	a	b	\$	A	S	S'
1	s7	s4		g6		
2			accept			
3	r(4)		r(4)			
4	s3			g5		
5	s8					
6			r(2)			
7	r(4)	s9	r(4)			
8			r(1)			
9			r(1)			

(b)

Terminals = {a, b, c, \$}
 Non-Terminals = {S, S', A, B}
 Rules = (0) S' → S\$
 = (1) S → ABA
 = (2) S → A
 = (3) A → aBB
 = (4) A → bc
 = (5) B → AAB
 = (6) B → cb
 Start Symbol = S'

	First	Follow
S'	a,b	
S	a,b	\$
A	a,b	a,b,c,\$
B	a,b,c	a, b,c, \$



	a	b	c	\$	A	B	S	S'
1	s4	s8			g5			
2				accept				
3				r(1)				
4	s4	s8			g10	g14		
5	s4	s8	s9	r(2)	g10	g6		
6	s4	s8			g3			
7	r(5)	r(5)	r(5)	r(5)				
8			s13					
9		s12						
10	s4	s8			g11			
11	s4	s8	s9		g11	g7		
12	r(6)	r(6)	r(6)	r(6)				
13	r(4)	r(4)	r(4)	r(4)				
14	s4	s8	s9		g10	g15		
15	r(3)	r(3)	r(3)	r(3)				

3. Given the following class definitions, and the local variable declarations in the function foo:

```

class classA {
    boolean x;
    int y;
}

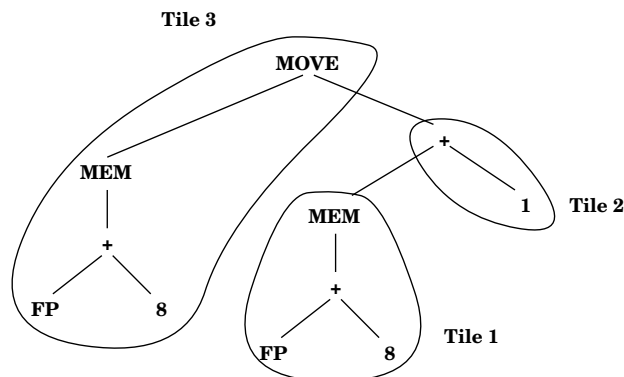
class classB {
    classA w[];
    int z;
}

int foo(int x,int y) {
    boolean w[];
    classA A;
    classB B;

    /* Body of foo */
}

```

Give the abstract assembly tree for each of the following statements, if they appeared in the body of foo. For each abstract assembly tree, tile the tree (by circling tiles on the tree), and then give the actual assembly, noting which code is associated with each tile.



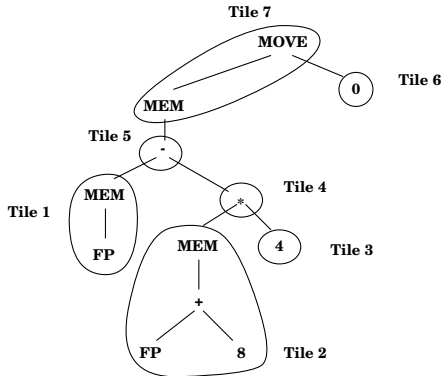
(a) `y++`;

```

lw  $ACC, 8($FP)      # Tile 1
addi $ACC, $ACC, 1    # Tile 2
SW  $ACC, 8($FP)     # Tile 3

```

(b) $w[y] = \text{false};$

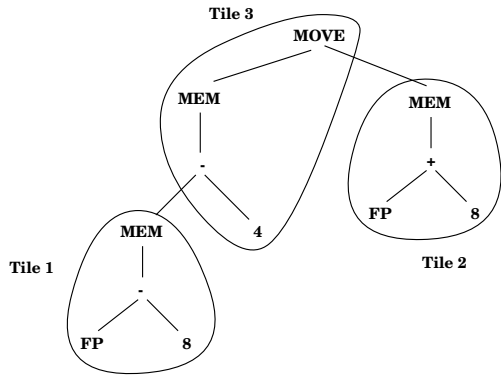


```

lw  $ACC, 0($FP)      # Tile 1
sw  $ACC, 0($ESP)     # Tile 5
addi $ESP, $ESP, -4   # Tile 5
lw  $ACC, 8(8)        # Tile 2
sw  $ACC, 0($ESP)     # Tile 4
addi $ESP, $ESP, -4   # Tile 4
addi $ACC, $zero, 4   # Tile 3
addi $ESP, $ESP, 4    # Tile 4
lw  $t1, 0($ESP)     # Tile 4
mul  $t1, $ACC         # Tile 4
mflo $ACC             # Tile 4
addi $ESP, $ESP, 4    # Tile 5
lw  $t1, 0($ESP)     # Tile 5
sub  $ACC, $t1, $ACC  # Tile 5
sw  $ACC, 0($ESP)     # Tile 7
addi $ESP, $ESP, -4   # Tile 7
addi $ACC, $zero      # Tile 6
addi $ESP, $ESP, 4    # Tile 7
lw  $t1, 0($ESP)     # Tile 7
sw  $ACC, 0($t1)     # Tile 7

```

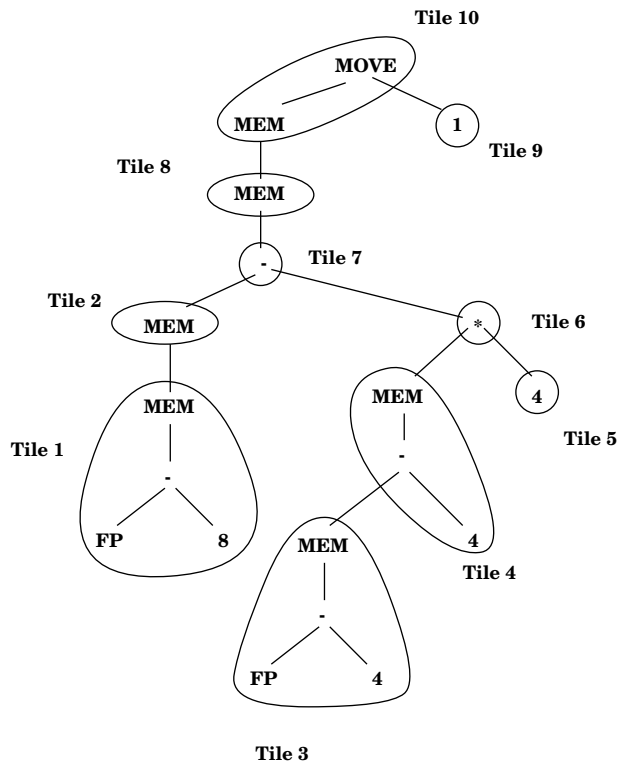
(c) $B.z = y;$



```

lw  $ACC, -8(FP)      # Tile 1
sw  $ACC, 0($ESP)    # Tile 3
addi $ESP, $ESP, -4  # Tile 3
lw  $ACC, 8($FP)     # Tile 2
addi $ESP, $ESP, 4   # Tile 3
lw  $t1, 0($ESP)     # Tile 3
sw  $ACC, -4($t1)    # Tile 3

```



(d) B.w[A.y].x = true;

```

lw  $ACC, -8(FP)      # Tile 1
lw  $ACC, 0($ACC)    # Tile 2
sw  $ACC, 0($ESP)    # Tile 7
addi $ESP, $ESP, -4  # Tile 7
lw  $ACC, -4($FP)    # Tile 3

```

```

lw  $ACC, -4($ACC)      # Tile 4
sw  $ACC, 0($ESP)      # Tile 6
addi $ESP, $ESP, -4    # Tile 6
addi $ACC, $zero, 4    # Tile 5
addi $ESP, $ESP, 4     # Tile 6
lw  $t1, 0($ESP)      # Tile 6
mul  $t1, $ACC         # Tile 6
mflo $ACC             # Tile 6
addi $ESP, $ESP, 4     # Tile 7
lw  $t1, 0($ESP)      # Tile 7
sub  $ACC, $t1, $ACC   # Tile 7
lw  $ACC, 0($ACC)     # Tile 8
sw  $ACC, 0($ESP)     # Tile 10
addi $ESP, $ESP, -4    # Tile 10
addi $ACC, $zero, 1    # Tile 9
addi $ESP, $ESP, 4     # Tile 10
lw  $t1, 0($ESP)     # Tile 10
sw  $ACC, 0($t1)      # Tile 10

```