Question 1 is in submitted file

Part 2 Question 1

LR(1) closure table			
Goto	Kernel	State	Closure
	{[S' -> .S, $]}	0	{[S' -> .S, $]; [S -> .a C, $]; [S -> .A C, $]; [A -> .a B b, c/d/a]; [A -> .A a, c/d/a]}
goto(0, S)	{[S' -> S., $]}	1	{[S' -> S., $]}
goto(0, a)	{[S -> a.C, $]; [A -> a.B b, c/d/a]}	2	{[S -> a.C, $]; [A -> a.B b, c/d/a]; [C -> .c C, $]; [C -> .d, $]; [B -> .b B, b]; [B -> .c C, b]}
goto(0, A)	{[S -> A.C, $]; [A -> A.a, c/d/a]}	3	{[S -> A.C, $]; [A -> A.a, c/d/a]; [C -> .c C, $]; [C -> .d, $]}
goto(2, C)	{[S -> a C., $]}	4	{[S -> a C., $]}
goto(2, B)	{[A -> a B.b, c/d/a]}	5	{[A -> a B.b, c/d/a]}
goto(2, c)	{[C -> c.C, $]; [B -> c.C, b]}	6	{[C -> c.C, $]; [B -> c.C, b]; [C -> .c C, $/b]; [C -> .d, $/b]}
goto(2, d)	{[C -> d., $]}	7	{[C -> d., $]}
goto(2, b)	{[B -> b.B, b]}	8	{[B -> b.B, b]; [B -> .b B, b]; [B -> .c C, b]}
goto(3, C)	{[S -> A C., $]}	9	{[S -> A C., $]}
goto(3, a)	{[A -> A a., c/d/a]}	10	{[A -> A a., c/d/a]}
goto(3, c)	{[C -> c.C, $]}	11	{[C -> c.C, $]; [C -> .c C, $]; [C -> .d, $]}
goto(3, d)	{[C -> d., $]}	7	
goto(5, b)	{[A -> a B b., c/d/a]}	12	{[A -> a B b., c/d/a]}
goto(6, C)	{[C -> c C., $]; [B -> c C., b]}	13	{[C -> c C., $]; [B -> c C., b]}
goto(6, c)	{[C -> c.C, $/b]}	14	{[C -> c.C, $/b]; [C -> .c C, $/b]; [C -> .d, $/b]}
goto(6, d)	{[C -> d., $/b]}	15	{[C -> d., $/b]}
goto(8, B)	{[B -> b B., b]}	16	{[B -> b B., b]}
goto(8, b)	{[B -> b.B, b]}	8	
goto(8, c)	{[B -> c.C, b]}	17	{[B -> c.C, b]; [C -> .c C, b]; [C -> .d, b]}
goto(11, C)	{[C -> c C., $]}	18	{[C -> c C., $]}
goto(11, c)	{[C -> c.C, $]}	11	
goto(11, d)	{[C -> d., $]}	7	
goto(14, C)	{[C -> c C., $/b]}	19	{[C -> c C., $/b]}
goto(14, c)	{[C -> c.C, $/b]}	14	
goto(14, d)	{[C -> d., $/b]}	15	
goto(17, C)	{[B -> c C., b]}	20	{[B -> c C., b]}
goto(17, c)	{[C -> c.C, b]}	21	{[C -> c.C, b]; [C -> .c C, b]; [C -> .d, b]}
goto(17, d)	{[C -> d., b]}	22	{[C -> d., b]}
goto(21, C)	{[C -> c C., b]}	23	{[C -> c C., b]}
goto(21, c)	{[C -> c.C, b]}	21	
goto(21, d)	{[C -> d., b]}	22	

LR PARSING TABLE

LR table										
State	ACTION					GOTO				
	a	b	c	d	$	S'	S	A	B	C
0	s2						1	3		
1					acc					
2		s8	s6	s7					5	4
3	s10		s11	s7						9
4					r_1					
5		s12								
6			s14	s15						13
7					r_8					
8		s8	s17						16	
9					r_2					
10	r_4		r_4	r_4						
11			s11	s7						18
12	r_3		r_3	r_3						
13		r_6			r_7					
14			s14	s15						19
15		r_8			r_8					
16		r_5								
17			s21	s22						20
18					r_7					
19		r_7			r_7					
20		r_6								
21			s21	s22						23
22		r_8								
23		r_7								

a) abbbccd

Input (tokens): a b b b c c d

Maximum number of steps: 100

PARSE

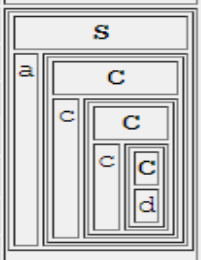
Trace				Tree
Step	Stack	Input	Action	
1	0	a b b b c c d $	s2	
2	0 a 2	b b b c c d $	s8	
3	0 a 2 b 8	b b c c d $	s8	
4	0 a 2 b 8 b 8	b c c d $	s8	
5	0 a 2 b 8 b 8 b 8	c c d $	s17	
6	0 a 2 b 8 b 8 b 8 c 17	c d $	s21	
7	0 a 2 b 8 b 8 b 8 c 17 c 21	d $	s22	
8	0 a 2 b 8 b 8 b 8 c 17 c 21 d 22	$		

b) accd

Input (tokens): a c c d

Maximum number of steps: 100

PARSE

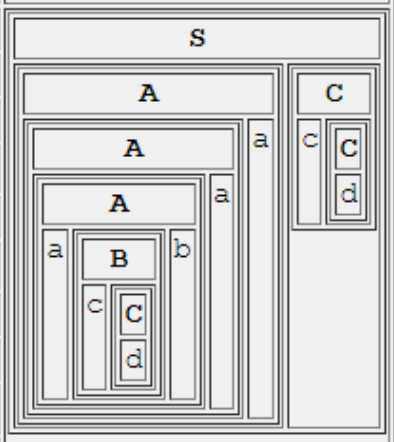
Trace				Tree
Step	Stack	Input	Action	
1	0	a c c d $	s2	
2	0 a 2	c c d $	s6	
3	0 a 2 c 6	c d $	s14	
4	0 a 2 c 6 c 14	d $	s15	
5	0 a 2 c 6 c 14 d 15	$	r_8	
6	0 a 2 c 6 c 14 C	$	19	
7	0 a 2 c 6 c 14 C 19	$	r_7	
8	0 a 2 c 6 C	$	13	
9	0 a 2 c 6 C 13	$	r_7	
10	0 a 2 C	$	4	
11	0 a 2 C 4	$	r_1	
12	0 S	$	1	
13	0 S 1	$	acc	

c) acdbaacd

Input (tokens): a c d b a a c d

Maximum number of steps: 100

PARSE

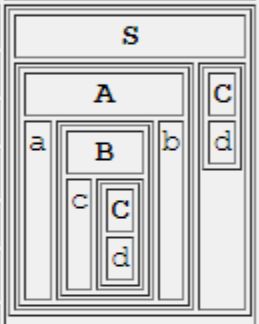
Trace				Tree
Step	Stack	Input	Action	
1	0	a c d b a a c d $	s2	
2	0 a 2	c d b a a c d $	s6	
3	0 a 2 c 6	d b a a c d $	s15	
4	0 a 2 c 6 d 15	b a a c d $	r_8	
5	0 a 2 c 6 C	b a a c d $	13	
6	0 a 2 c 6 C 13	b a a c d $	r_6	
7	0 a 2 B	b a a c d $	5	
8	0 a 2 B 5	b a a c d $	s12	
9	0 a 2 B 5 b 12	a a c d $	r_3	
10	0 A	a a c d $	3	
11	0 A 3	a a c d $	s10	
12	0 A 3 a 10	a c d $	r_4	
13	0 A	a c d $	3	
14	0 A 3	a c d $	s10	
15	0 A 3 a 10	c d $	r_4	
16	0 A	c d $	3	
17	0 A 3	c d $	s11	
18	0 A 3 c 11	d $	s7	
19	0 A 3 c 11 d 7	$	r_8	
20	0 A 3 c 11 C	$	18	
21	0 A 3 c 11 C 18	$	r_7	
22	0 A 3 C	$	9	
23	0 A 3 C 9	$	r_2	
24	0 S	$	1	
25	0 S 1	$	acc	

d) acdbd

Input (tokens): a c d b d

Maximum number of steps: 100

PARSE

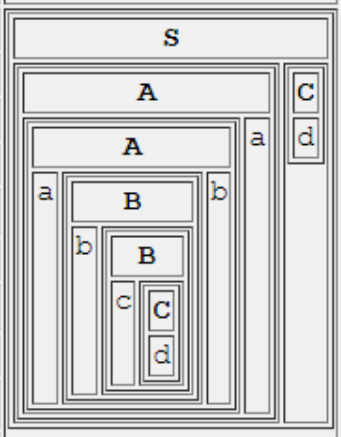
Trace				Tree
Step	Stack	Input	Action	
1	0	a c d b d $	s2	
2	0 a 2	c d b d $	s6	
3	0 a 2 c 6	d b d $	s15	
4	0 a 2 c 6 d 15	b d $	r_8	
5	0 a 2 c 6 C	b d $	13	
6	0 a 2 c 6 C 13	b d $	r_6	
7	0 a 2 B	b d $	5	
8	0 a 2 B 5	b d $	s12	
9	0 a 2 B 5 b 12	d $	r_3	
10	0 A	d $	3	
11	0 A 3	d $	s7	
12	0 A 3 d 7	$	r_8	
13	0 A 3 C	$	9	
14	0 A 3 C 9	$	r_2	
15	0 S	$	1	
16	0 S 1	$	acc	

e) abcdbad

Input (tokens): a b c d b a d

Maximum number of steps: 100

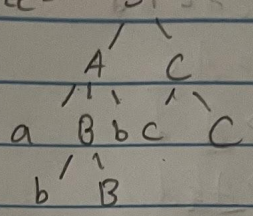
PARSE

Trace				Tree
Step	Stack	Input	Action	
1	0	a b c d b a d $	s2	
2	0 a 2	b c d b a d $	s8	
3	0 a 2 b 8	c d b a d $	s17	
4	0 a 2 b 8 c 17	d b a d $	s22	
5	0 a 2 b 8 c 17 d 22	b a d $	r_8	
6	0 a 2 b 8 c 17 C	b a d $	20	
7	0 a 2 b 8 c 17 C 20	b a d $	r_6	
8	0 a 2 b 8 B	b a d $	16	
9	0 a 2 b 8 B 16	b a d $	r_5	
10	0 a 2 B	b a d $	5	
11	0 a 2 B 5	b a d $	s12	
12	0 a 2 B 5 b 12	a d $	r_3	
13	0 A	a d $	3	
14	0 A 3	a d $	s10	
15	0 A 3 a 10	d $	r_4	
16	0 A	d $	3	
17	0 A 3	d $	s7	
18	0 A 3 d 7	$	r_8	
19	0 A 3 C	$	9	
20	0 A 3 C 9	$	r_2	
21	0 S	$	1	
22	0 S 1	$	acc	

Part 2 Question 3

a) $S \xrightarrow{rm} AC$
$\rightarrow AdC$
$\rightarrow aBbcC$
$\rightarrow abBbcC$

Parse Tree -



Handle - bB
Phrases - abBbcC, abBb, cC, bB
Simple phases - bB, cC

b) accCd

$S \xrightarrow{rm} aC$
$\rightarrow acC$
$\rightarrow accC$

$S \xrightarrow{rm} AC$
$\rightarrow AcC$
$\rightarrow AccC$
$\rightarrow AcccC$

does not work
you will always get 3 c's before C

c) aCbaacd

$S \xrightarrow{rm} AC$
$\rightarrow AcC$
$\rightarrow Acd$
$\rightarrow Aacd$
$\rightarrow Aaacd$
$\rightarrow aBbaacd$
$\rightarrow aCbaacd$ x

does not work
you will have a c before C

d) acdabd

$S \xrightarrow{rm} AC$
$\rightarrow Ad$
$\rightarrow Aad$
$\rightarrow aBbad$
$\rightarrow acdbd$

$S \xrightarrow{rm} AC$
$\rightarrow Ad$
$\rightarrow aBbd$
$\rightarrow abBbd$
$\rightarrow abcCbd$
$\rightarrow abcdbd$

Does not work either a won't be before bd or you won't have bd at end

e) abCbad

$S \xrightarrow{rm} AC$
$\rightarrow Ad$
$\rightarrow Aad$
$\rightarrow aBbad$
$\rightarrow abBbad$
$\rightarrow abcCbad$
$\rightarrow$

$\rightarrow acCbad$
$\rightarrow abbBbad$

Does not work because you are missing a b above C or have a extra C before aC