Answer all questions in the exam book. You may keep the exam questions after you are done.

1. [56] Consider the following grammar G:
   1. S → X a ⊥
   2. X → Y b
   3. X → Y Z c
   4. Y → d
   5. Z → b

For each of the following grammar classes, if G is of that class, give the appropriate parsing table. If it is not of that class, fully explain why it isn't.

(a) LL(0)
(b) LL(1)
(c) LR(0)
(d) LR(1)
(e) SLR(1)
(f) LALR(1)
(g) General precedence

(a) An LL(0) grammar generates a single string. This grammar generates 2 strings so can't be LL(0).
(b) First(Yb) = First(Yzc) = {d} so can't be LL(1).
(c) Try to generate LR(0) table first:

S0	[S→.Xa⊥] [X→.Yb] [X→.YZc] [Y→.d]	X → 1 Y→ 2 d→3
S1	S→X.a⊥]	a→4
S2	[X→Y.b] [X→Y.Zc] [Z→.b]	b→5 Z→6
S3	[Y→d.]	Reduce 4
S4	S→Xa.⊥]	Reduce 1
S5	[X→Yb.] [Z→b.]	

Reduce – reduce conflict so not LR(0).

(d) LR(1)

S0	[S→.Xa,⊥] [X→.Yb,a] [X→.YZc,a] [Y→.d,b]	X → 1 Y→ 2 d→3
S1	S→X.a,⊥]	a→4
S2	[X→Y.b,a] [X→Y.Zc,a] [Z→.b,c]	b→5 Z→6
S3	[Y→d.,b]	Reduce 4
S4	S→Xa.,⊥]	Reduce 1
S5	[X→Yb.,a] [Z→b.,c]	a→ Reduce 2 c→ Reduce 5
S6	[X→YZ.c,a]	c→7
S7	[X→YZc.,a]	Reduce 3

No conflicts so LR(1).

(e) SLR(1)

Copy over table from (c) -- LR(0)

S0	[S→.Xa⊥] [X→.Yb] [X→.YZc] [Y→.d]	X → 1 Y→ 2 d→3
S1	S→X.a⊥]	a→4
S2	[X→Y.b] [X→Y.Zc] [Z→.b]	b→5 Z→6
S3	[Y→d.]	Reduce 4
S4	S→Xa.⊥]	Reduce 1
S5	[X→Yb.] [Z→b.]	a→Reduce 2 c→Reduce 5

In S5: Follow(X) ={a}
Follow(Z) = {c}
Disjoint so fix up above table and continue

S6	[X→YZ.c]	c→7
S7	[X→YZc.]	Reduce 3

No further conflicts so SLR(1)

(f) LALR(1) – ALL item lists in (e) (LR(1)) are disjoint, so grammar is LALR(1) by default.

(g) Generate precedence relations:

From rule 2:
Y = b
From rule 3:
Y = Z
Z = c
Y < First(Z) = b
Thus Y=b and Y<b. Precedence relations are not unique, so not precedence grammar.

2. [8] Is the **language** given by the grammar in problem above LL(1)? Prove your answer.

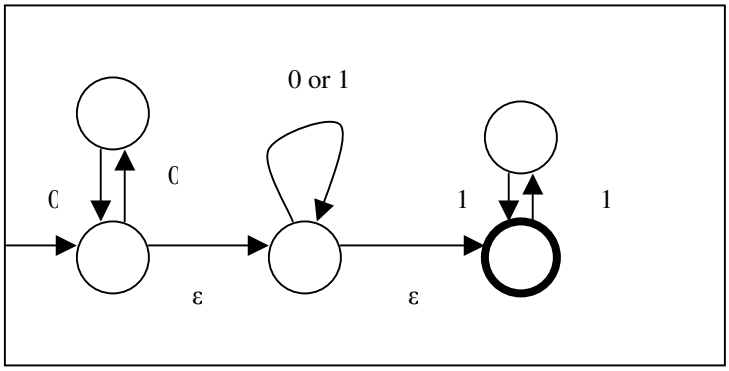
Yes LL(1). Grammar only generates 2 strings: dba⊥ and dbca⊥. Write grammar as follows:

1. S → dbX ⊥
2. X→a
3. X → ca

Language generated by this grammar is same as original grammar and it is LL(1) since
First(a) = {a}, First(ca)={c} and
First(a) ∩ First(ca) = Ø

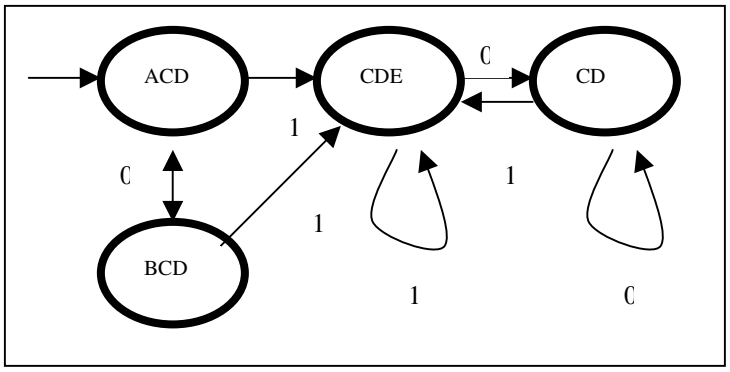
3. [12] For the regular expression
(00)* (0 | 1)* (11)*

(a) Give the non-deterministic FSA that accepts the same set.

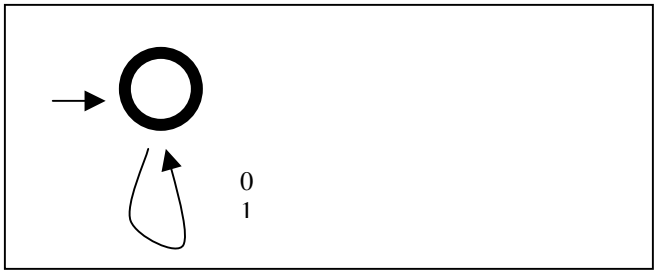


(b) Give the minimal state DFA that recognizes the same set.

NFA→DFA algorithm gives:



Which reduces to:



(c) Give a regular grammar that recognizes the same set.

1. S → ε | 0S | 1S

4[8] Show that the language A^nB^{2n} for n>0 is LR(k). What is K?

A simple PDA recognizes this set. Push A on stack. When find first B, pop one A for every 2 Bs read. End of string and empty stack occur at same time.

K=0 here since no lookahead is needed.

(b) Show that $A^nB^{2n}C^n$ for n>0 is not LR(k) for any k.

No PDA can work, therefore it can't be LR(K) for any K. To show this, a PDA must stack and match As to Bs and no way to count the Cs or must stack As and Bs and match Bs to Cs and no way to count the As.

5. [12] Answer each of the following:

(a) Assuming the usual grammar for expressions, give the Polish Postfix for the expression: (3+4*(2+7)+8*5)

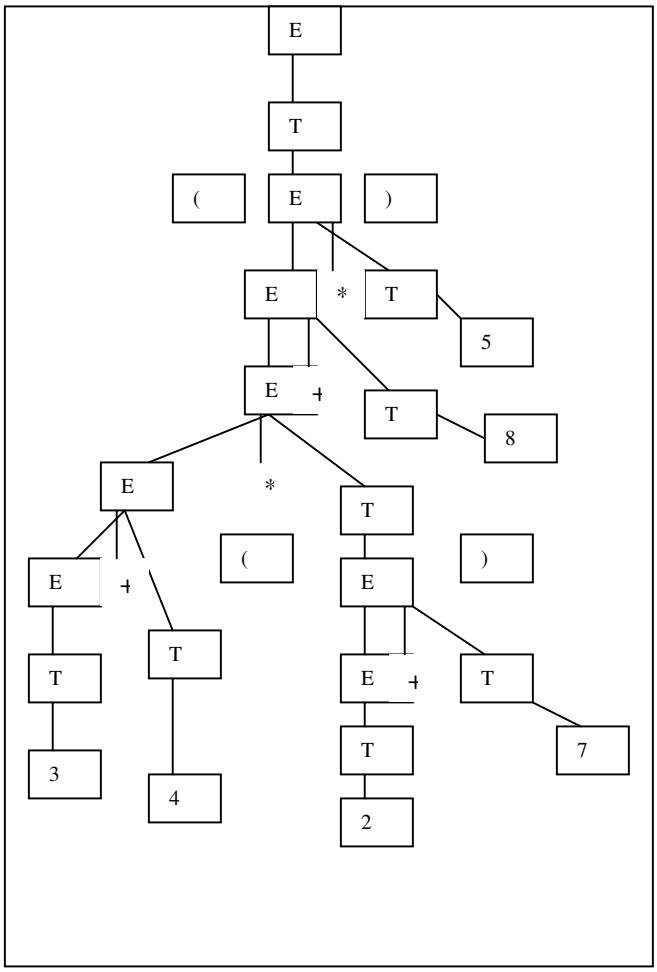
3427+*+85*+

(b) If the grammar for an expression is given by the grammar:

E → E + T | E – T | E * T | E / T | T

T → number | (E)

Give the parse tree for the expression in (a).



(c) Using the grammar in (b), give the Postfix for the expression in (a).

34+27+*8+5*

(d) Give a set of quads that would be the output of a parser for the expression in (a).

<add, 2, 7, T1>
<mpy, 4, T1,T2>
<add, 3,T2,T3>
<mpy,8,5,T4>
<add,T3,T4,T5>

6. [4] Why do you want only synthesized attributes in YACC? What would happen if you defined YACC with inherited attributes?

YACC works bottom up and inherited attributes come top down. Yacc would fail since value of attributes would not be known at point of production reduction.