CSE431 Midterm Examination

Name: KEY

ID Number: _____

- Show your work to receive maximum credit. Partial credit will be given, but only if work is shown.
- Circle or otherwise indicate your answer to each question.
- Please keep your written answers brief and to the point. Incorrect or rambling statements can hurt your score on a question.
- If a question requires you to provide code, please use Java syntax. Minor syntax errors will not be counted against you, but extremely sloppy code will result in deductions.
- This exam is worth 20% of your final grade.
- The exam is broken up into 5 parts and is worth 100 points. It is your responsibility to make sure you have all of the pages.
- You must sign the pledge below for your exam to count. Any cheating will cause the students involved to receive an F for this course. Other action may be taken.

Section	Possible Points	Score
I	20	
II	15	
III	20	
IV	25	
V	20	
Total	100	

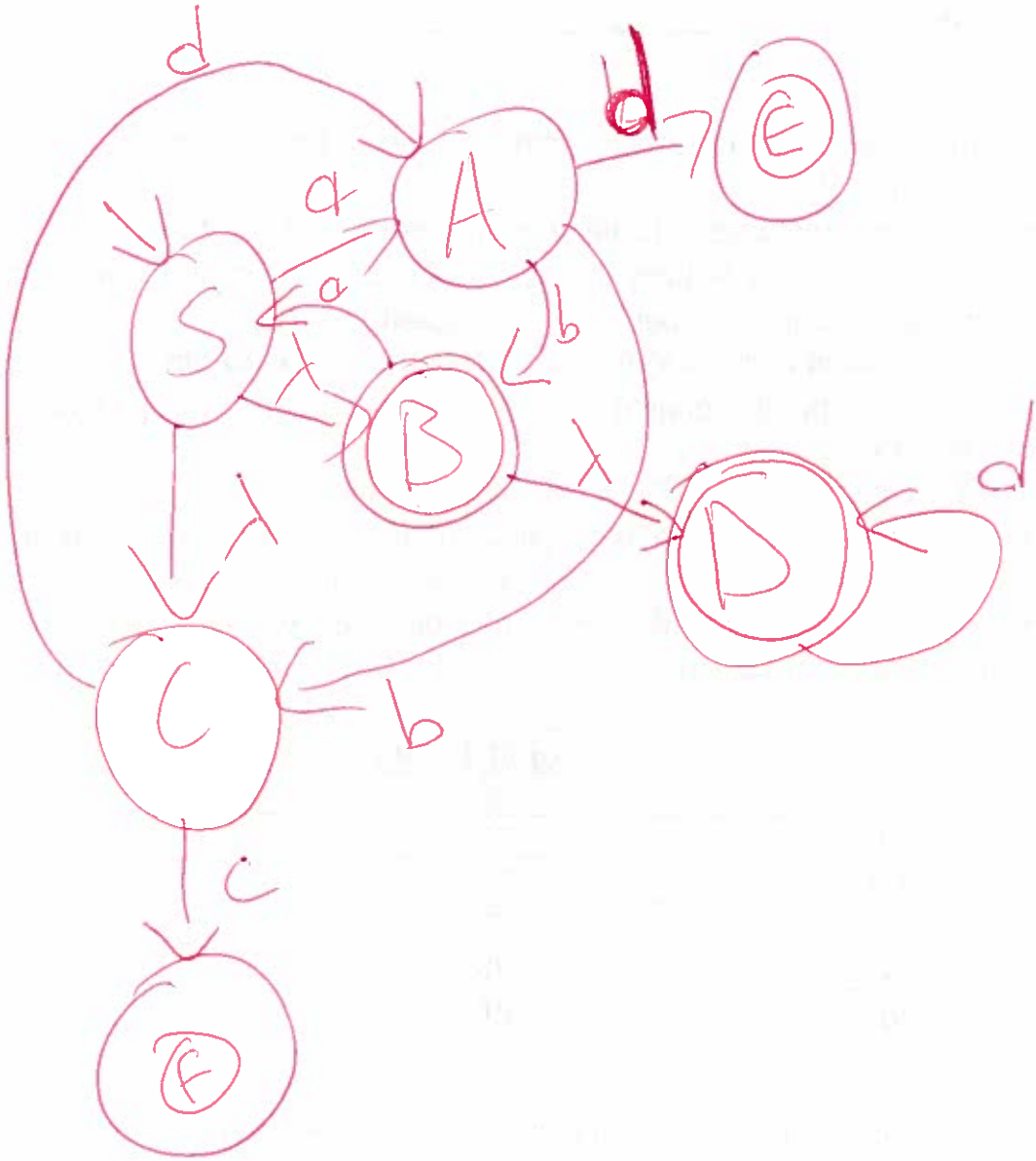
Pledge: On my honor, I have neither given nor received any unauthorized aid on this exam.

Signed: _____

1. (20 points) Answer the following:

(a) (10 points) Create an FSA from the following grammar. Do not take any shortcuts – create the FSA just as the algorithm would, with every production clearly and explicitly included. Hint: your result should be non-deterministic.

S -> aA
| B
| C
A -> d
| bB
| bC
B -> aS
| λ
| D
C -> dA
| c
D -> dD
| λ



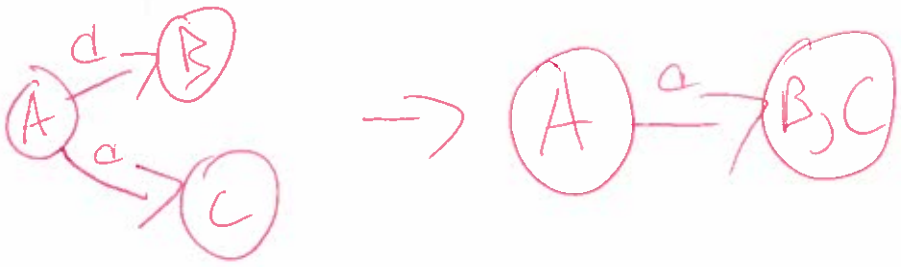
(b) (10 points) Remove the non-determinism from the NFA you created in the previous question. You do not need to show the DFA, creating a table of states is sufficient.

State	a	b	c	d
SBD	ASBD	—	—	D
ASBD	ASBD	BDC	—	DE
D	—	—	—	D
BDC	SBD	—	F	DA
DE	—	—	—	D
F	—	—	—	—
DA	—	BDC	—	D

2. (15 points) For the following questions, you must explain your answers. If the answer is false, give a counter example showing why. If the answer is true, write a couple of sentences explaining your answer, giving as much detail as possible.

(a) (5 points) True or false: If the non-determinism is removed from an NFA, the resulting DFA will always have more states than the original NFA.

False



(b) (5 points) True or false: If a grammar is SLR(1), it is also LALR(1).

True. LALR is a more powerful technique than SLR, so if a grammar can be parsed with SLR, it can also be parsed with LALR.

(c) (5 points) True or False: LL(1) grammars cannot be ambiguous.

True. LL(1) grammars cannot contain ambiguity as that would lead to overlap in the predict sets.

3. (20 points) Answer the following:

(a) (15 points) Construct the Derives-λ, FIRST, FOLLOW, and PREDICT sets for the following grammar:

S -> ClassDef $
ClassDef -> id { Stmts }
Stmts -> Stmt Stmts
| λ
Stmt -> id (Params)
| type id
Params -> Param Params
| λ
Param -> id

Non-Terminal	Derives - λ	First	Follow
S	F	id	—
ClassDef	F	id	$
Stmts	T	id, type	}
Stmt	F	id, type	id, type, }
Params	T	) id	)
Param	F	id	id,)

Production	Predict
S -> ClassDef $	id
ClassDef -> id { Stmts }	id
Stmts -> Stmt Stmts	id, type
Stmts -> λ	}
Stmt -> id (Params)	id
Stmt -> type id	type
Params -> Param Params	id
Params -> λ	)
Param -> id	id

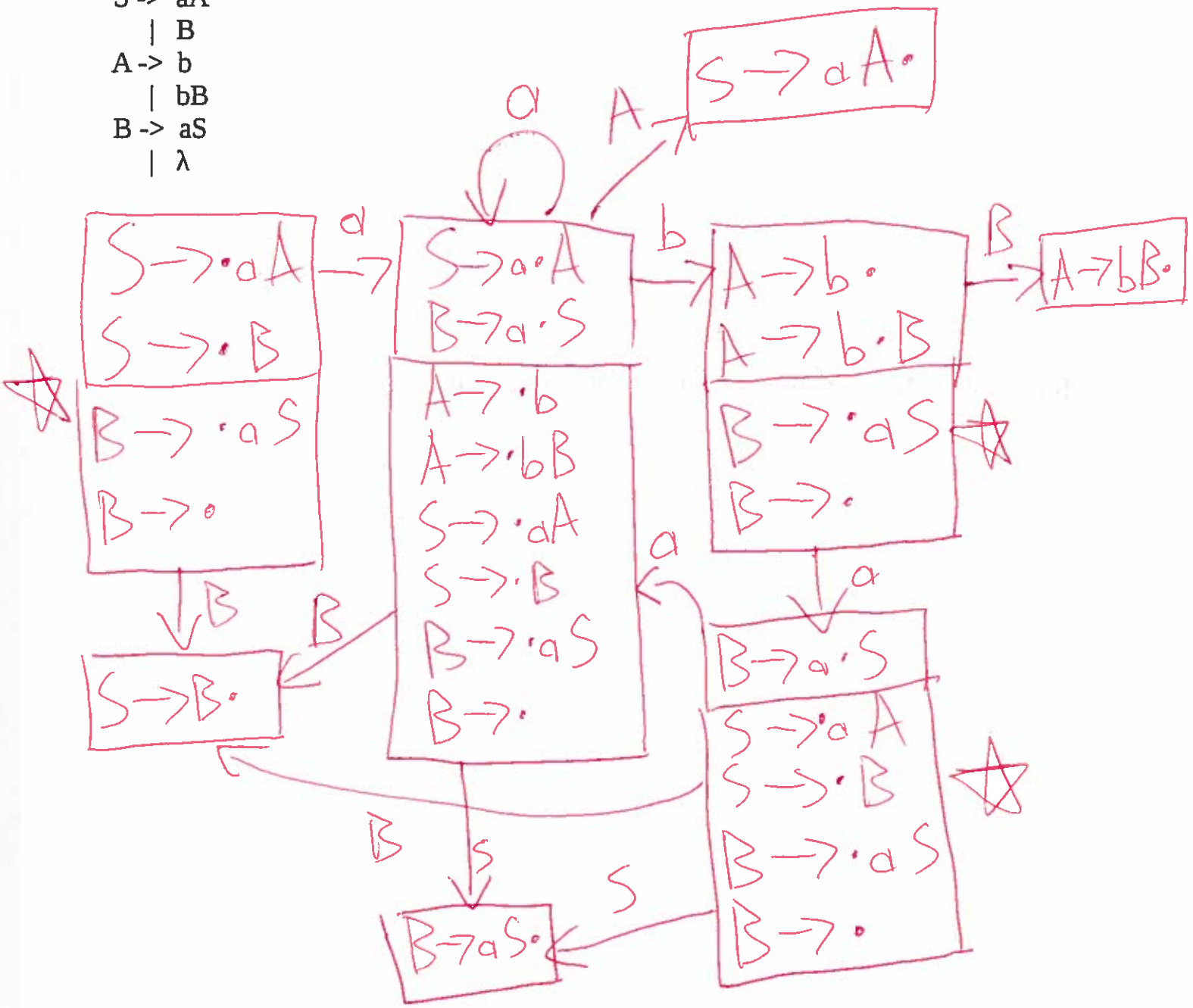
(b) (5 points) Is this grammar LL(1)? You may assume that it is not ambiguous.

This grammar is LL(1). There is no left recursion, no common prefix, and the predict sets do not overlap.

4. (25 points) Answer the following:

(a) (15 points) Construct the CFSM for the following grammar:

S -> aA
  | B
A -> b
  | bB
B -> aS
  | λ



(b) (5 points) Are there any conflicts in this grammar? If there are, state what type of conflicts they are and mark the state(s) that they appear in.

Conflicts are marked with a ☆. All conflicts are shift-reduce conflicts,

(c) (5 points) Is this grammar LR(0)? Explain how you know.

It is not LR(0), because LR(0) cannot resolve conflicts,

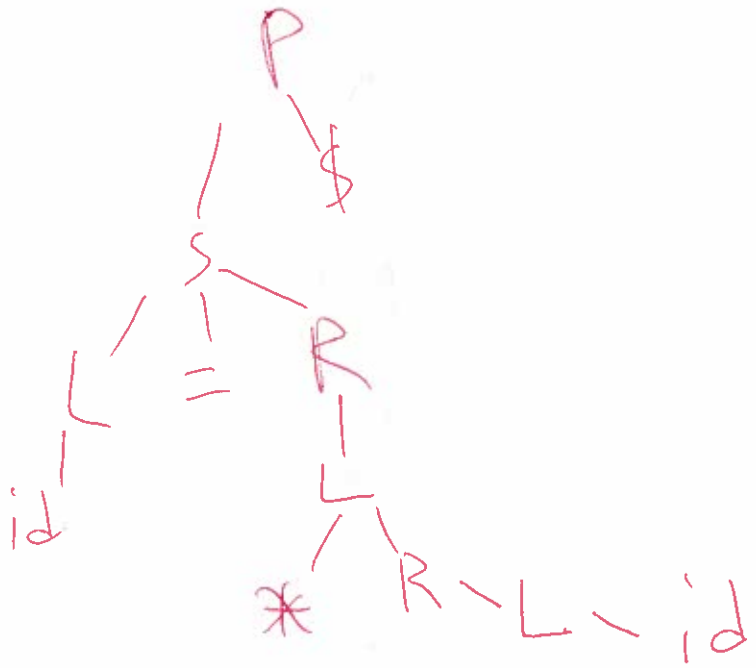
5. (20 points) You are given the following:

(1) P -> S$
(2) S -> L = R
(3) | R
(4) L -> id
(5) | * R
(6) R -> L

In the table below, underlined values represent shifts, non-underlined values represent reductions:

State	$	=	Id	*	P	S	L	R
1	6		4	5	Accept	1	2	3
2	6	7						
3	3							
4	4	4						
5			4	5			9	8
6	1							
7			4	5			9	8
8	5	5						
9	6	6						
10	2							

(a) (5 points) Construct a parse tree for the string "id = * id $"



(b) (15 points) Show the bottom up parse process for the string "id = * id $". You must show the state of the stack and input stream at every step to receive full credit.

