Key Solution

King Saud University
College of Computer and Information Sciences
Computer Science Department

Final Exam

Academic Year: 2016/2017
Second Semester

BSc Program
Course Name/No. : Programming Language Compilation / CS340

Exam Date: 17 /5/2017:

Total Number of Pages: 7 pages (including this cover page)

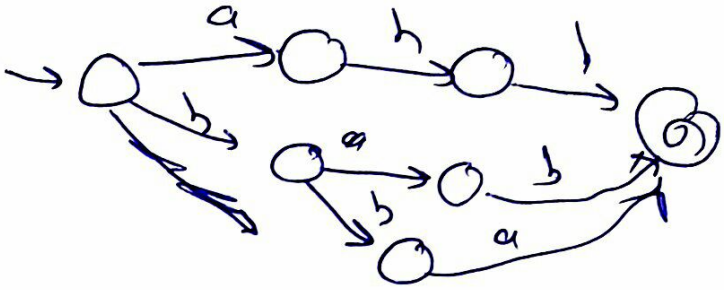
Student Name	
Student ID.	

Exercise No.	Full Mark	Student Mark
1.	6	
2.	4	
3.	8	
4.	9	
5.	5	
6.	8	
Total	40	

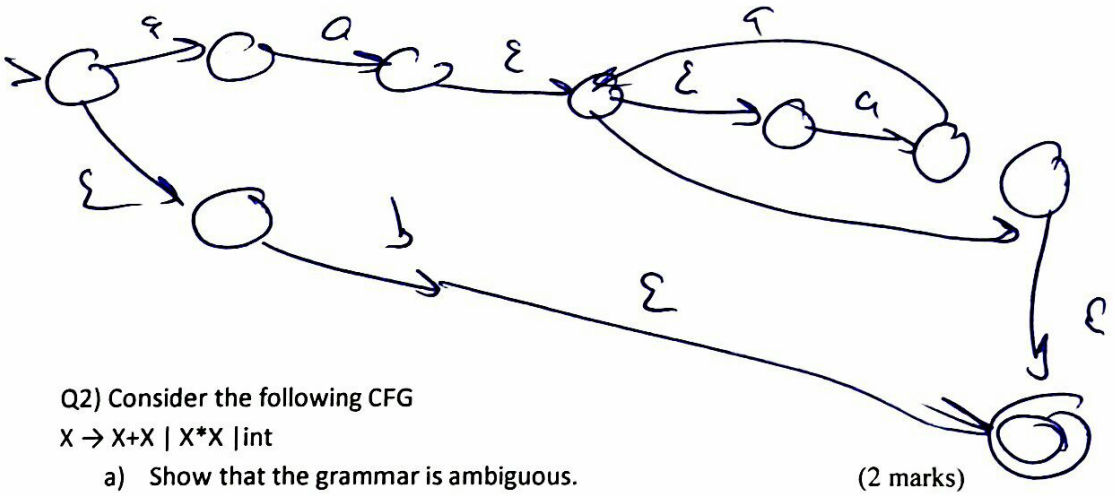
Q1) a) Write regular expressions for a language over the alphabet $\Sigma = \{a,b\}$ that accepts all strings that begin and end with a different character. (2 marks)

$a(a+b)^* b + b(a+b)^* a$

b) Design a DFA, over the alphabet $\Sigma = \{a,b\}$, that accepts a string if contains exactly two **b**'s and one **a**. (2 marks)



c) Convert the regular expression **(aa*)+b** into an NFA (2 marks)

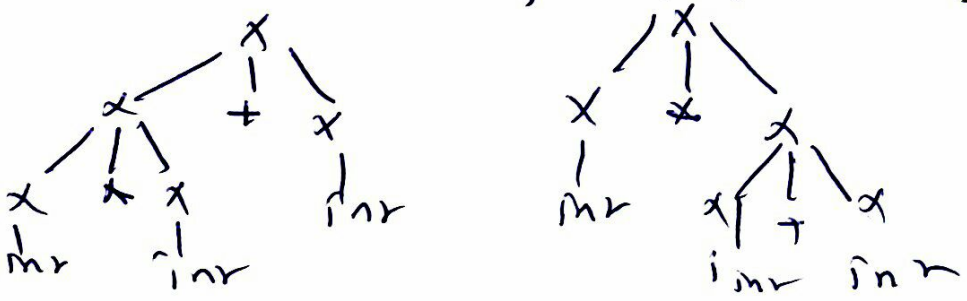


Q2) Consider the following CFG

$X \rightarrow X+X \mid X*X \mid int$

a) Show that the grammar is ambiguous. (2 marks)

Since int * int + int has the following 2 parse trees, the grammar is ambiguous



b) Rewrite the grammar so that + has left associativity and * has right associativity. (2 marks)

$X \rightarrow X+nr \mid nr*X \mid nr$

Q3) Consider the following grammar over the alphabet $\sum$ = {u,v,w,x,y,z};

S → UVW

U → u | Wv | ϵ

V → w | xU | ϵ

W → y | z | ϵ

a) Complete the missing entries in the following table that gives the first and follow sets for each terminal and non-terminal symbols: (4 marks)

Symbol	First	Follow
S	{u, y, z, v, w, x, ϵ}	{$}
U	{u, y, z, v, ϵ}	{v, x, y, z, $}
V	{w, x, ϵ}	{y, z, $}
W	{y, z, ϵ}	{$, v}
u	{u}	{u, x, y, z, $}
v	{v}	{v, x, y, z, $}
w	{w}	{y, z, $}
x	{x}	{u, y, z, v, $}
y	{y}	{$, v}
z	{z}	{$, v}

b) Draw the corresponding LL(1) table. (3 marks)

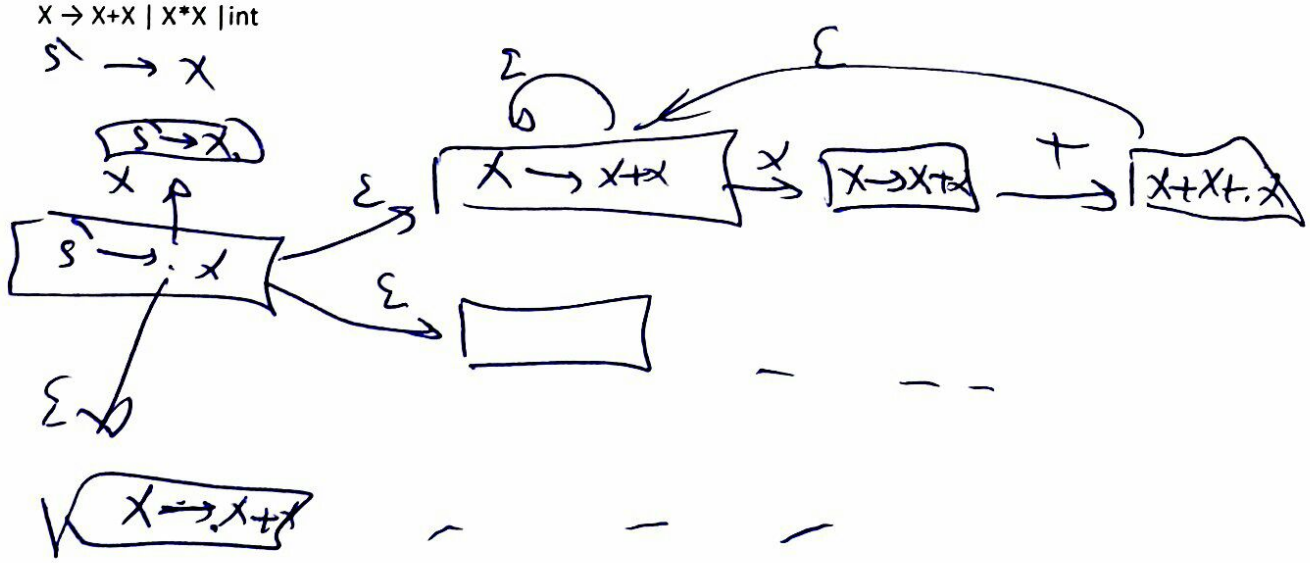
	int	+	*	X	$
S	X → X+X, X → X*X, X → int				
X	X → X+X, X → X*X, X → int				

b) Is the grammar an LL(1) grammar? Why? or why not? (1 marks)

it is not because we have a multiply defined entry

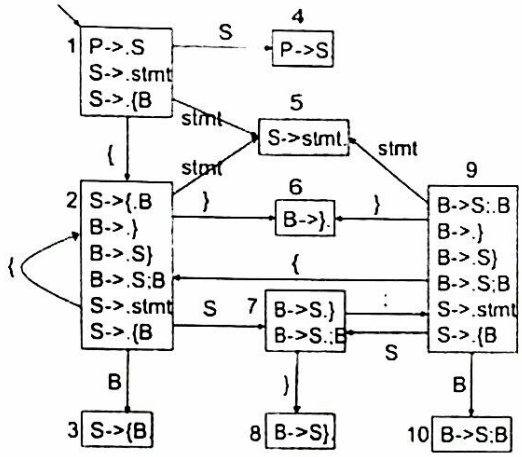
Q4) A) Draw an NDF that recognizes all valid items for the grammar (3 marks)

X → X+X | X*X | int



c) Is that above grammar SLR grammar? Why? or why not? (2 marks)

D) Consider the following DFA for viable prefixes of a grammar **(4 marks)**



Assuming that an SLR parser resolves shift-reduce conflicts by choosing to shift, show the operation of such a parser on the input string **{stmt;;}**. Write your answer as a table of a suitable format and make it clear if the statement is accepted or rejected.

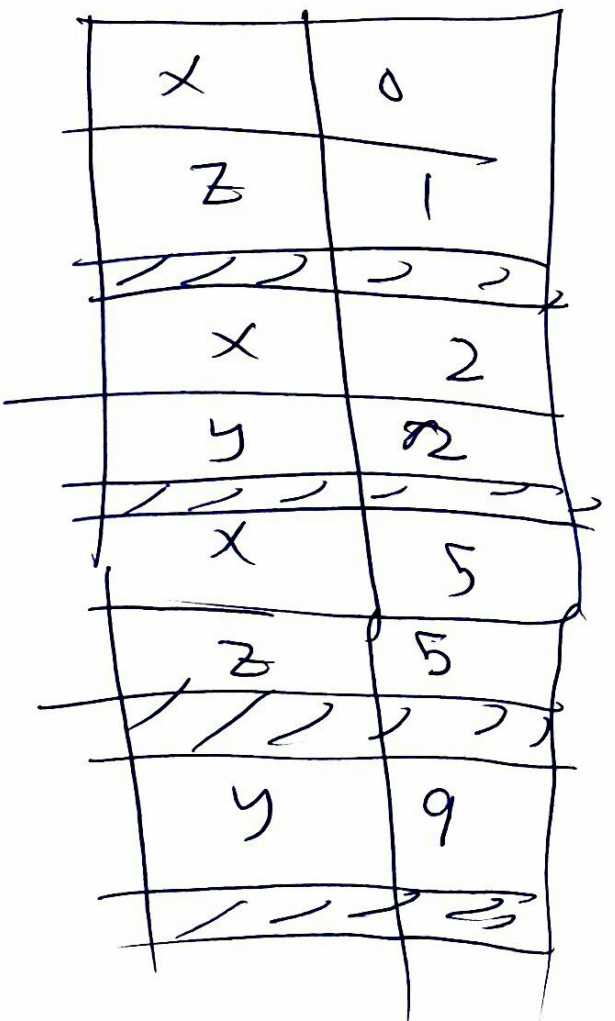
Configuration	DFA	Action
|{stmt;;}$	1	Shift
{|stmt;;}$	2	Shift
{stmt|;;}$	5	reduce S→stmt
{S|;;}$		

Q5) Consider the following program

```
0:  int x = 137;
1:  int z = 42;
2:  int MyFunction(int x, int y)
3:    printf("%d, %d, %d\n", x, y, z);
4:  {
5:      int x, z;
6:      z = y;
7:      x = z;
8:      {
9:          int y = x;
10:         {
11:             printf("%d, %d, %d\n", x, y, z);
12:         }
13:         printf("%d, %d, %d\n", x, y, z);
14:     }
15:     printf("%d, %d, %d\n", x, y, z);
16: }
17:
```

1) Mark each variable with the line number it was declared in, by writing above it @ followed by the line number. (3 marks)

2) Draw the symbol table that you have used when you reached line 13. (2 marks)



Q6) A) Write a code generator for the language constructs If e1=e2 then e3 else e4
(4 marks)

```
cgen(if e1=e2 then e3 else e4)
cgen(e1)
sw   $a0  0($sp)
addiu $sp $sp -4
cgen(e2)
lw   $t1  4($sp)
addiu $sp $sp 4
beq  $a0  $t1  true_branch
false_branch:
  cgen(e4)
  b end_if
true_branch:
  cgen(e3)
end_if:
```

B) Write the code that would be produced by cgen for the expression **x+5**. You can assume that x is the only variable in the procedure. (4 marks)

```
lw $a0  4($fp)
sw $a0  0($sp)
addiu $sp $sp -4)
li $a0  5
lw $t1  (4($sp)
add $a0 $t1 $a0
addiu $sp $sp 4
```