CS_218 COMPILERS

(Attempt 2 questions out of 3)

Question 1.

(a) Consider a language which consists of an *assignment* symbol **:=**, the four mathematical symbols **-**, **+**, *****, **/**, and *identifiers* formed by an alphabetic character followed by nought or more alphanumeric characters. Write a *Lex* file which recognises and returns **id**, **plus**, **minus**, **mul**, **div**, or **assign** appropriately. (You may assume that the tokens have been declared elsewhere). Your program should contain a **main** function which calls the parser and prints nothing to the screen except the number of lines in the input file.

[8 marks]

(b) Remove the ϵ productions from the following grammar G_1 in such a way that the language of the resulting grammar $L(G_2) = L(G_1)$:-
$A \rightarrow Ba|b$
$B \rightarrow c|CdD$
$C \rightarrow \epsilon|eAD$
$D \rightarrow \epsilon|f$

[4 marks]

(c) Explain the important aspects of the *semantical analysis* phase of an *analysis-synthesis* compiler. **[4 marks]**

(d) Create *first* and *follow* sets for the following grammar :-
$E \rightarrow TE'$
$E' \rightarrow +TE'|\epsilon$
$T \rightarrow FT'$
$T' \rightarrow *FT'|\epsilon$
$F \rightarrow (E)|id$

[5 marks]

(e) State, in words and mathematically, the two disjunctions necessary for a grammar to be LL(1).
[4 marks]

Question 2.

(a) Draw a DFA state transition diagram which recognises all strings containing the substring ab when the alphabet $\Sigma = \{a, b, c\}$ **[5 marks]**

(b) Construct the LR state set for the following augmented grammar :-
$E' \rightarrow E$
$E \rightarrow E + T|T$
$T \rightarrow T * F|F$
$F \rightarrow (E)|id$

[8 marks]

(c) Is the grammar in (b) above LR(0). Give detailed explainations for your answer.

[5 marks]

(d) Given the following LL(1) parsing table show a trace of the input string **id + id * id**.

	id	+	*	(	)	$
E	$E \rightarrow TE'$			$E \rightarrow TE'$		
E'		$E' \rightarrow +TE'$			$E' \rightarrow \epsilon$	$E' \rightarrow \epsilon$
T	$T \rightarrow FT'$			$T \rightarrow FT'$		
T'		$T' \rightarrow \epsilon$	$T' \rightarrow *FT'$		$T' \rightarrow \epsilon$	$T' \rightarrow \epsilon$
F	$F \rightarrow id$			$F \rightarrow (E)$		

[7 marks]

Question 3.

(a) Remove immediate left recursion from the following grammar :-

$E \rightarrow E + T|T$
$T \rightarrow T * F|F$
$F \rightarrow (E)|id$

[5 marks]

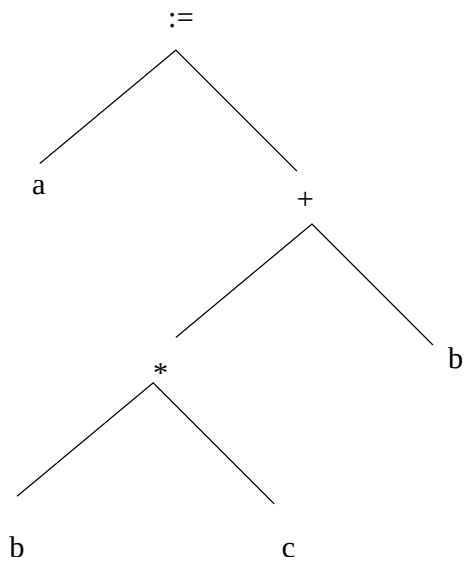
(b) Given the following LR(0) grammar...
$S \rightarrow bSe$
$S \rightarrow a$
...and its item state set...

0	$S' \rightarrow \cdot S, S \rightarrow \cdot bSe, S \rightarrow \cdot a$
1	$S' \rightarrow S\cdot$
2	$S \rightarrow b \cdot Se, S \rightarrow \cdot bSe, S \rightarrow \cdot a$
3	$S \rightarrow bS \cdot e$
4	$S \rightarrow a\cdot$
5	$S \rightarrow bSe\cdot$

...construct the LR(0) parse table. **[5 marks]**

(c) Let G = (T, N, S, P) be a *context-free* grammar. Describe each of the four components of G. **[4 marks]**

(d) Given the following code fragment...
a := b * c + b
...and the following tree...



..show what **Three Address Code** might be generated by a compiler. **[5 marks]**

(e) Draw and carefully label a diagram representing the *analysis-synthesis* model of a compiler. Show on your diagram the parts which are considered as the *front* and *back* ends. Explain why a compiler is usually divided into front and back ends. **[6 marks]**