PRIFYSGOL CYMRU; UNIVERSITY OF WALES

DEGREE EXAMINATIONS MAY/JUNE 2003

SWANSEA

Computer Science

CS 218 Compilers

Attempt 2 questions out of 3

Time allowed: 2 hours

Students are permitted to use the dictionaries provided by the University

Students are NOT permitted to use calculators

CS_218 COMPILERS

(Attempt 2 questions out of 3)

Question 1.

(a) Explain the important aspects of the *syntactical analysis* phase of an *analysis-synthesis* compiler. **[5 marks]**

(b) Draw a DFA State Transition Diagram which will recognise all strings which include the substring *be* if the alphabet $\Sigma = \{b, e, g\}$ **[5 marks]**

(c) Remove immediate left recursion from the following grammar :

$exp \rightarrow exp\ addop\ term | term$
$term \rightarrow term\ mulop\ factor | factor$
$factor \rightarrow (exp) | id$

[5 marks]

(d) Consider the following LL(1) parsing table :

	id	−	∗	(	)	$
A	A → CB			A → CB		
B		B → − CB			B → ϵ	B → ϵ
C	C → ED			C → ED		
D		D → ϵ	D → ∗ED		D → ϵ	D → ϵ
E	E → *id*			E →(A)		

Write down a trace of a parse of the string $(id - id) * id$ using the table. (Assume A is the start symbol). **[5 marks]**

(e) State the subset relationships for the following types of language :
SLR(1), LALR(1),LR(0), LR(1). Explain the meaning of the *LR* and the integer. **[5 marks]**

Question 2.

(a) Draw and label a diagram showing the structure of the *analysis-synthesis* model of a compiler. Indicate on your diagram which parts are considered to be the *front end* and *back end.* Explain why a compiler is usually divided into *front* and *back* ends.

[5 marks]

(b) For a language whose alphabet $\Sigma = \{i, t, e\}$ write a regular expression for all strings which do not contain the substring te

[5 marks]

(c) Explain why we would want to *left factor* grammar rules using the following fragment to demonstrate :

$if\ exp \rightarrow if\ con\ then\ exp\ else\ exp\ fi | if\ con\ then\ exp\ fi$

[5 marks]

(d) Compute *first* and *follow* sets for the following LL(1) grammar : $C \rightarrow cD|Ed$
$D \rightarrow EFC|e$
$E \rightarrow \epsilon|f$
$F \rightarrow g$

[5 marks]

(e) State the conditions which must hold for all item states in order that a grammar is LR(0). Explain the two types of violation of this condition and the conflict in each case.

[5 marks]

Question 3.

(a) Consider a programming language which contains just the assignment symbol ":=" and identifiers formed from an alphabetic character followed by a series of alphanumeric characters. Draw and label a state transition diagram which could form the basis of a lexical analyser for the language. You may assume the existence of the following tokens : ident, assign, error, endOfInput.

Label your diagram to show when each token should be returned. **[5 marks]**

(b) Use the following production rules and corresponding *first* and *follow* sets for a LL(1) grammar to create a parsing table :

$E \rightarrow TE^{'}$
$E^{'} \rightarrow +TE^{'}|\epsilon$
$T \rightarrow FT^{'}$
$T^{'} \rightarrow *FT^{'}|\epsilon$
$F \rightarrow (E)|id$

$First(E) = \{(, id\}$
$First(T) = \{(, id\}$
$First(F) = \{(, id\}$
$First(E^{'}) = \{+, \epsilon\}$
$First(T^{'}) = \{*, \epsilon\}$
$Follow(E) = \{), \$\}$
$Follow(E^{'}) = \{), \$\}$
$Follow(T) = \{+,), \$\}$
$Follow(T^{'}) = \{+,), \$\}$
$Follow(F) = \{+, *,), \$\}$

[6 marks]

(c) Construct the LR(0) state set for the following augmented grammar :

$S^{'} \rightarrow S$
$S \rightarrow (S)S|\epsilon$

[6 marks]

(d) Remove the ϵ productions from the following grammar :-

$R \rightarrow Aa|b$
$A \rightarrow c|BdD$
$B \rightarrow \epsilon|eRD$
$D \rightarrow \epsilon|f$

[4 marks]

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