

DUBLIN CITY UNIVERSITY

SEMESTER ONE EXAMINATIONS 2008

MODULE: Compiler Construction 1
(Title & Code) CA448

COURSE: B.Sc. in Computer Applications (SE)

YEAR: 4

EXAMINERS: Dr. F. Bannister
Dr. P. Gibson
Dr. G. Hamilton (ext. 5017)

TIME ALLOWED: 2 Hours

INSTRUCTIONS: Please answer ALL questions.
All questions carry equal marks

Requirements for this paper
Please tick (X) as appropriate

- | | |
|--------------------------|------------------------------|
| <input type="checkbox"/> | <i>Log Table</i> |
| <input type="checkbox"/> | <i>Graph Paper</i> |
| <input type="checkbox"/> | <i>Attached Answer Sheet</i> |
| <input type="checkbox"/> | <i>Statistical Tables</i> |
| <input type="checkbox"/> | <i>Floppy Disk</i> |
| <input type="checkbox"/> | <i>Actuarial Tables</i> |

**THE USE OF PROGRAMMABLE OR TEXT STORING
CALCULATORS IS EXPRESSLY FORBIDDEN**

**PLEASE DO NOT TURN OVER THIS PAGE UNTIL YOU ARE
INSTRUCTED TO DO SO**

1. Construct a non-deterministic finite state automaton (NFA) which recognises the same language as defined by the following regular expression: [10 marks]

$$(ab)^*a(ba)^*$$

2. Using the subset construction method, convert your NFA in Question 1 into a corresponding deterministic finite state automaton (DFA), showing clearly the start state and accepting states. [10 marks]

3. Prove that the following grammar for regular expressions over the alphabet $\{0,1\}$ is ambiguous: [10 marks]

$$\begin{aligned} R &\rightarrow R \mid R \quad (\text{alternation}) \\ R &\rightarrow R R \quad (\text{concatenation}) \\ R &\rightarrow R^* \quad (\text{Kleene closure}) \\ R &\rightarrow (R) \\ R &\rightarrow 0 \\ R &\rightarrow 1 \end{aligned}$$

by giving two different parse trees for the following expression:

$$(0 \mid 1 0)^*$$

4. Construct an unambiguous grammar which describes the same language as the grammar in question 3, and which deals with the precedence of the operators (Kleene closure has the highest precedence, followed by concatenation, followed by alternation). [10 marks]

5. Determine whether or not the following grammar is LL(1): [10 marks]

$$\begin{aligned} S &\rightarrow ABC \\ A &\rightarrow bA \\ A &\rightarrow \epsilon \\ B &\rightarrow aB \\ B &\rightarrow \epsilon \\ C &\rightarrow b \end{aligned}$$

6. Determine whether or not the following grammar is LR(0): [10 marks]

$$\begin{aligned} S &\rightarrow Ab \\ S &\rightarrow B \\ A &\rightarrow aB \\ B &\rightarrow a \\ B &\rightarrow aA \end{aligned}$$

7. Determine whether or not the following grammar is LR(1): [10 marks]

$S \rightarrow Aa$
 $S \rightarrow Bb$
 $S \rightarrow cC$
 $A \rightarrow D$
 $B \rightarrow D$
 $C \rightarrow Ab$
 $C \rightarrow Ba$
 $D \rightarrow \epsilon$

8. Determine whether or not the grammar in Question 7 is LALR(1). [10 marks]

9. Consider the following grammar for Boolean expressions: [10 marks]

$E \rightarrow E \text{ or } T$
 $E \rightarrow T$
 $T \rightarrow T \text{ and } F$
 $T \rightarrow F$
 $F \rightarrow \text{not } F$
 $F \rightarrow \text{true}$
 $F \rightarrow \text{false}$
 $F \rightarrow (E)$

Add attributes and rules that calculate the Boolean value of an expression (hint: the only attribute that will be required for each non-terminal will be one denoting its Boolean value).

10. Construct a directed acyclic graph for the following code fragment which identifies all common sub-expressions. [10 marks]

$G := C * (A + B) + (A + B);$
 $C := A + B;$
 $A := (C * D) + (E - F);$

[Total marks: 100]