THE OPEN UNIVERSITY OF SRI LANKA
THE FACULTY OF ENGINEERING TECHNOLOGY
THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ECX 6235 – Compiler Design

056

Final Examination – 2007 / 2008 ***(Closed Book Type)***

Date : Friday, 2nd of May, 2008 **Time : 09:30 – 12:30**

INSTRUCTIONS TO CANDIDATES

You must answer any two questions from Part A and another two questions from Part B. Questions 1 – 3 carry 30 marks each and question 4 – 7 carry 20 marks each.

Part A

Consider following description of a compiler to answer questions 1 and 2.

A compiler converts expressions of simple arithmetic operations into their pseudo-Assembler code equivalents in the following way:

$C = 15$	⇒	EQU Acc, #15d
$C = A + B$	⇒	ADD R_A, R_B ;
$C = A - B$	⇒	SUB R_A, R_B ;
$C = A \times B$	⇒	MUL R_A, R_B ;
$C = A \div B$	⇒	DIV R_A, R_B ;

Where A, B and C are variables that can carry any decimal number and R_A, R_B denote two registers that are used when performing the arithmetic operations in the pseudo-Assembler code. Assume that all the results of Assembler operations will end up in the Accumulator (Acc). The numbers in the pseudo-Assembler code must start with hyphen (#) mark and have letter "d" to indicate the numbers are in decimal. The input (i.e. the left column of expressions) can be given as algebraic expressions, like

$$A = 5; \qquad B = 6; \qquad C = A + B.$$

Or they can be given as numerical equations, like

$$C = 5 + 6.$$

Every line of the input can carry only one expression. Assume all input numeral are whole numbers (no decimal point).

Your task is to build the part of the compiler that performs only the above mentioned conversions of the code.

Note: Write down all your assumptions where necessary, when answering questions 1 and 2 below.

1. (a) What is the alphabet of this compiler? [4 marks]
   (b) Give the token table for this lexical analyzer. [6 marks]
   (c) What are the functions of the lexical analyzer in the above compiler? [4 marks]
   (d) Write an algorithm for the above lexical analyzer in pseudo code. [16 marks]

2. (a) What are the functions of the syntax analyzer for the above compiler? [2 marks]
   (b) Write the set of rules of the syntax analyzer. [7 marks]
   (c) Write an algorithm for the syntax analyzer in pseudo code. [21 marks]

3. (a) Strings are input into the Turing Machine on an unlimited tape. The string consists only of 1s and 0s. The Turing Machine converts every third digit into its opposite, i.e. 1 to 0 and 0 to 1. 'B' denotes a blank symbol. You may neglect any possible error occurrences. Assume that the length of the string can be only of multiples of three, i.e. 3, 6, 9, …, and that there are at least two blank symbols between the strings. The tape will consist of five strings at least, but there may be more. If the Turing Machine will encounter a point (.) symbol it will terminate its operation. Note that the point (.) symbol can only be encountered instead of a string.

For example consider following input and output strings:

Input: … B B B 0 1 $\underline{0}$ 1 0 $\underline{1}$ 1 0 $\underline{1}$ 1 0 $\underline{0}$ B B B …

Output: … B B B 0 1 $\underline{1}$ 1 0 $\underline{0}$ 1 0 $\underline{0}$ 1 0 $\underline{1}$ B B B …

(Digits to be changed and the changed ones are underlined.)

Draw the transition graph for the above Turing Machine. The labels in the transition graph must be in the following format: [input symbol, output symbol, operation]. [20 marks]

(b) Give the transition table for the transition graph drawn above. The labels in the transition table must be in the following format: [next state, input symbol, operation]. [10 marks]

Part B

4. (a) Find the NDFA equivalent to **ba* (b+c)** $^+$**ca**$^+$. [5 marks

(b) Convert the above NDFA to equivalent DFA. [10 marks

(c) Give the formal definition of the regular expressions. [5 marks

5. (a) Define the closure of Language L. [6 marks

(b) Give the formal definition of derivation tree. [8 marks

(c) Give the formal definition of LR(0) Grammar. [6 marks

6. (a) A grammar G is defined by the following productions

$$\begin{aligned}
S &\rightarrow e\,f\,D\,N\,a \\
L &\rightarrow m \mid C\,d\,f \mid N\,b\,C\,a\,m \\
C &\rightarrow b\,m\,N \mid f\,d \mid N \\
D &\rightarrow S\,a\,C \mid n \mid C \\
N &\rightarrow b\,a \mid C\,m\,D
\end{aligned}$$

Find equivalent grammar without useless symbols and unit productions. [8 marks

(b) Convert above into Chomsky Normal Form (CNF). [12 marks

7. (a) A grammar G is given by

$$G = (\{S, A, B\}, \{a, b, c\}, P, S)$$

where P is given by the following productions:

$$\begin{aligned}
S &\rightarrow aAbc \\
Ab &\rightarrow bA \\
Ac &\rightarrow Bbcc \\
bB &\rightarrow Bb \\
aB &\rightarrow aaA \\
aB &\rightarrow aa \\
S &\rightarrow abc
\end{aligned}$$

Show that this grammar generates the language

$$L = \{ a^n b^n c^n \mid n \geq 1\}.$$

[12 mark

(b) Is L context- free? Prove your answer. [8 mark