00047

The Open University of Sri Lanka
Faculty of Engineering Technology
Department of Electrical & Computer Engineering

Study Programme	: Bachelor of Software Engineering Honours
Name of the Examination	: Final Examination
Course Code and Title	**: EEX6563/ECX6263 Software Construction**
Academic Year	: 2019/20
Date	: 11th October 2020
Time	: 1330 – 1630 hrs
Duration	: **3 hours**

General Instructions

1. Read all instructions carefully before answering the questions.
2. This question paper consists of **Five (5)** questions in **Four (4)** pages.
3. Answer **all** questions in **Part A** and any **TWO** questions from **Part B**.
4. Answer for each question should commence from a new page.
5. This is a Closed Book Test (**CBT**).
6. Answers should be in clear handwriting.
7. Do not use Red colour pen.
8. Clearly state your **assumptions**, if any.

Part A – Answer all questions

[Q1]. Suppose you are designing a FCTC (Flow chart to C) compiler which gives C language code equivalent to the flow chart as shown in figure 1.1. Refer the example FCTC compiler shown in Figure 1.2 to answer **[Q1]**.

Figure 1.1: The FCTC Compiler

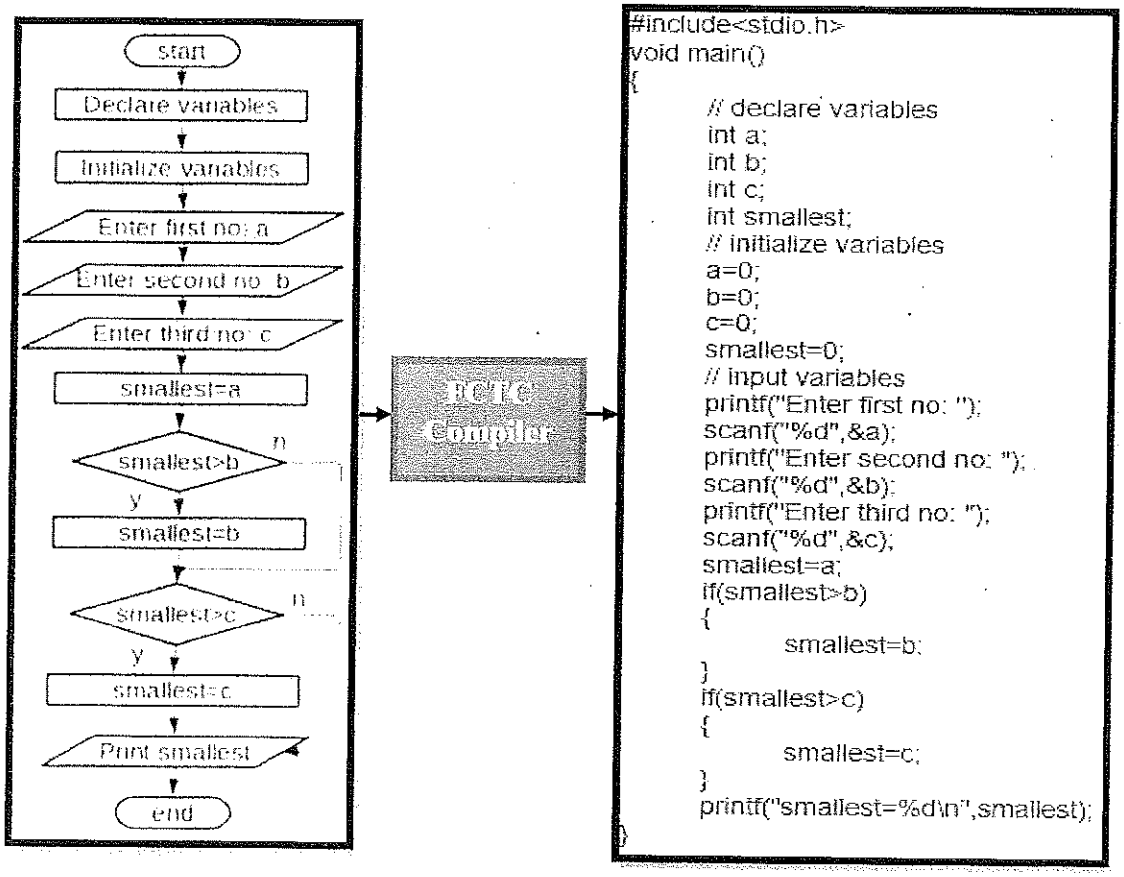

Figure 1.2: An example of the FCTC Compiler

a) Briefly explain the four types of grammars with applications. **[10 marks]**
b) Explain the compilation phases of the FCTC compiler. You must clearly show the input and output of each phase. **[30 marks]**
c) Define the grammar **G** for the FCTC compiler showing the start symbol, the production rules, terminals, and non-terminals. **[10 marks]**
d) Write LEX implementation syntax for the FCTC compiler. **[10 marks]**

Part B – Answer ANY TWO questions

[Q2].

a) Consider the following grammar for Boolean expressions (E is a Non terminal).

E -> E or E | E and E

E -> not E

E -> (E)

E -> true | false

E -> ID

(i) Show that this grammar is ambiguous by using the string: **not(ID and ID or ID) and ID** **[04 Marks]**

(ii) Rewrite the grammar to remove the ambiguity. Make sure that your received grammar accepts the same language as the original. **[04 Marks]**

(iii) Then derive the above string in (a) using your grammar. **[04 Marks]**

b) Consider the following productions.

$U \rightarrow xT \mid yTP \mid \varepsilon$

$T \rightarrow yxU \mid xyP$

$P \rightarrow xxU \mid yyT \mid \varepsilon$

(i) Identify the type of this grammar and write the grammar in the form G = (N, T, P, S). **[04 Marks]**

(ii) Derive the "**yxyyyyxxxxxyxx**" strings using this grammar. **[04 Marks]**

[Q3].

a) Give regular expressions to generate the following strings over the alphabet {0, 1}.

(i). Set of all string where every 01 is followed by at least one 1.

(ii). Set of all strings containing only 3 1's.

(iii). Set of all strings of length 3 or more and starting with 10.

[06 Marks]

b) Find the NDFA equivalent to the regular expression **(a + b) (ab + ba)* bb** **[06 Marks]**

c) Convert the NDFA constructed in question 3) b) to equivalent DFA. **[08 Marks]**

[Q4]. A grammar G is given by

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

where P is given by the following productions:

$S \rightarrow aAbc$

$Ab \rightarrow bA$

$Ac \rightarrow Bbcc$

$bB \rightarrow Bb$

$aB \rightarrow aaA$

$aB \rightarrow aa$

$S \rightarrow abc$

(i) Define formal grammar G using standard notations. **[04 marks]**

(ii) Show that this grammar generates the language

$L = \{a^n b^n c^n \mid n \geq 1\}$. **[10 marks]**

(iii) Is L context-free? State the reasons for your answer. **[06 marks]**

[Q5]. Consider the grammar rules are given below (VendingMachine, STOCK, SALES, ITEM, SALE are non-terminals and others are terminals).

VendingMachine → stock STOCK sales SALES
STOCK → STOCK ITEM | ε
ITEM → item price qty
SALES → SALES SALE | ε
SALE → item price

(a) Derive the string: ***stock item price qty sales item price*** **[02 Marks]**

(b) Define the Chomsky Normal Form (CNF) for CFGs. **[04 Marks]**

(c) Convert the given grammar into CNF. **[12 Marks]**

(d) Derive the above string in (a) using new grammar in (c) **[02 Marks]**