Please

END TERM EXAMINATION

SIXTH SEMESTER [B.TECH./M.TECH.] MAY-JUNE-2015

Paper Code: IT308	Subject: Compiler Design
Time : 3 Hours	Maximum Marks :60

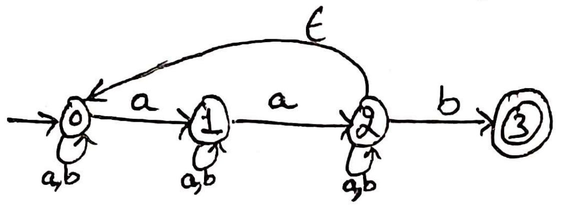
Note: Attempt any five questions.

Q1 (a) What is lexical analyser? How it interacts with Parser? What are the roles of a lexical analyser? **(5)**

(b) Which tool is used to generate lexical analyser? Explain its working in detail. **(5)**

(c) What are the difficulties faced during lexical analysis of the code? **(2)**

Q2 (a) Define Finite Automata. Convert the given NFA to DFA and give regular expression for the language accepted by this NFA. **(4)**



(b) Explain vital functions and features of phases of a compiler with an example. How symbol table is generated and managed? **(6)**

(c) Explain sources of code optimization. **(2)**

Q3 (a) Differentiate between top-down parsing and bottom up parsing. **(3)**

(b) What are shift reduce parsers? Give example of such parsers. Show the processing of two string "cdd" and "ccd" for the given shift reduce parser step by step. **(9)**

Given Grammar: S→CC C→cC C→d

Parsing table:

States	Action			GOTO	
	c	d	$	S	C
0	S3	S4		1	2
1			accept		
2	S3	S4			5
3	S3	S4			6
4	R3	R3	R3		
5			R1		
6	R2	R2	R2		

Q4 (a) Consider the context free grammar: $S \to SS+|SS*|a$ and the string "aa+a*".

(i) Give a leftmost derivation for the string. **(1)**

(ii) Give a rightmost derivation for the string. **(1)**

(iii) Give a parse tree for the string. **(1)**

(iv) Is the grammar ambiguous or unambiguous? Justify your answer. **(2)**

(v) Describe the language generated by this grammar. **(1)**

(b) Show that the following grammar is LALR(1) but not SLR(1).

$S \to Aa|bAc|dc|bda$

A→d. **(6)**

P.T.O.

IT-308

Q5 (a) Consider the following grammar and answer the questions below:- **(2x4=8)**

rexpr→rexpr + rterm |rterm
rterm→rterm rfactor|rfactor
rfactor→rfactor* | rprimary
rprimary→a|b

(i) Left factor this grammar.
(ii) Does left factoring makes the grammar suitable for top-down parsing? Justify your answer.
(iii)In addition to left factoring, eliminate left recursion from the original grammar.
(iv)Is the resulting grammar suitable for top-down parsing? Justify your answer.

(b) Show that the following grammar is SLR(1) but not LL(1): S→SA|A **(4)**
A→a

Q6 (a) For the given SDD, give syntax tree, annotated parse trees and dependency graph for "int a, b, c" this expression:- **(4)**

	Productions	**Sematic rules**
1	D→TL	L.inh=T.type
2	T→int	T.type=integer
3	T→float	T.type=float
4	L→L_1,id	L_1.inh=L.inh addType(id.entry, L.inh)
5	L→id	addType(id.entry, L.inh)

(b) Define SDD and type of SDD. How evaluation of attributes is done? **(3+2)**
(c) Explain run time storage organization and data structures used. **(3)**

Q7 (a) Why intermediate code representation is required? What do you understand by low level representation? Explain all intermediate representation. **(8)**
(b) Translate arithmetic expression a+-(b+c) into all low level intermediate code representations. **(4)**

Q8 Attempt **any four** of the following:- **(3x4=12)**
(a) Code generators with code generation algorithm.
(b) Issues in the design of code generator.
(c) Register allocation.
(d) YACC specification and which parser does it generate.
(e) How to set precedence and associability in YACC.

****

IT-308