Code No: R31051 **R10** **Set No: 1**

III B.Tech. I Semester Supplementary Examinations, June/July - 2014

COMPILER DESIGN

(Computer Science and Engineering)

Time: 3 Hours **Max Marks: 75**

Answer any FIVE Questions
All Questions carry equal marks
*****

1. What are the various phases of the compiler? Explain each phase in detail. Write down the output of each phase for the expression a:=b+c*50.

2. (a) Explain the role of the lexical analyser.
   (b) Give the regular expression to recognize
   (i) Identifiers such that they start with the alphabet
   (ii) Float number in exponent form.
   (c) Differentiate between compilers and interpreters.

3. (a) Consider the grammar.
   bexpr → bexpr or bterm | bterm
   bterm → bterm and bfactor | bfactor
   bfactor → not bfactor | (bexpr) | true | false
   Construct the predictive parser table for the above grammar and parse tree for the sentence not (true or false).
   (b) Why should we use Regular Expressions to define the lexical syntax of a language?

4. (a) Compute LR(0) items for the following grammar:
   S→L=R | R
   L→ *R | id
   R→L
   (b) Mention the conflicts that occur during shift-reduce parsing.

5. (a) Write and explain an algorithm for constructing LALR Parser table.
   (b) Construct the LALR Parsing table for the following grammar:
   E→E + T | T
   T→T * F | F
   F → (E)/id

6. (a) Explain how an L-attributed grammar is converted into a translation scheme.
   (b) Explain the differences between static and dynamic storage allocation schemes.

7. (a) List and explain various intermediate code forms with an example.
   (b) Explain dead code elimination, strength reduction and loop optimization techniques.

8. Explain how data flow equations are set up and solved. Explain different notation of data flow analysis.

*****

Code No: R31051 **R10** **Set No: 2**

III B.Tech. I Semester Supplementary Examinations, June/July - 2014

COMPILER DESIGN

(Computer Science and Engineering)

Time: 3 Hours **Max Marks: 75**

Answer any FIVE Questions

All Questions carry equal marks

*****

1. (a) What is a pre-processor? What are the various functions that are performed by a Pre-processor?
(b) Give the structure of a compiler and also mention functions performed by a compiler.

2. (a) Give the reasons for separating the lexical analysis from syntax analysis?
(b) Write short notes on the language for specifying lexical analyzer.
(c) Differentiate between the terms: Token, Pattern and Lexeme.

3. (a) What are the pre-processing steps required for predictive parsing?
(b) Remove the left recursion for the following grammar and construct predictive parsing table.

$S \rightarrow iEtSS' \mid a$
$S' \rightarrow eS \mid \varepsilon$
$E \rightarrow b$

4. (a) Construct the LR Parsing table for the following grammar:

$E \rightarrow E + T \mid T$
$T \rightarrow T * F \mid F$
$F \rightarrow (E)/id$

(b) How bottom-up parsers differs from top-down parsers?

5. (a) Compare & contrast SLR with LALR. Define Kernel items & Non-Kernel items.
(b) Show the following grammar is LALR (1)

$S \rightarrow Aa \mid bAc \mid dc \mid bda$
$A \rightarrow d$

6. (a) Write the syntax directed definitions for constructing syntax tree for an expression and construct the syntax tree for an expression a – 4 + c.
(b) Explain various storage allocation strategies with its merits and demerits.

7. (a) Construct the DAG for the expression: a + a * (b - c) + (b - c) * d.
(b) Explain various machine independent code optimization techniques.

8. (a) Explain the process of register allocation using graph coloring using an example.

*****

R10 **Set No: 3**

Code No: R31051

III B.Tech. I Semester Supplementary Examinations, June/July - 2014

COMPILER DESIGN

(Computer Science and Engineering)

Time: 3 Hours **Max Marks: 75**

Answer any FIVE Questions
All Questions carry equal marks
*****

1. (a) Explain various phases of a compiler with an example.
   (b) Write a short note on cross-compilers.

2. (a) What is a Regular Expression? List and explain the rules that define the regular expression.
   (b) Construct DFA for the Regular Expression (a/b)*abb(a/b)*. Explain the procedure to construct DFA.

3. (a) Construct LL(1) parser table for the following grammar:
   E→E+T
   T→T+F
   F→(E) | id
   (b) Explain the model of a non-recursive predictive parser with an example.

4. (a) Consider the ambiguous grammar & Construct the collection of sets of LR(0) items.
   S→AS | b
   A→SA | a
   (b) Explain the various actions performed by shift-reduce parsers with an example.

5. (a) Write and explain the algorithm for LALR parsing technique .
   (b) Construct CLR parsing table for the following grammar:
   S→L=R | R
   L→ *R | id
   R→L

6. (a) Construct the syntax directed definition to convert infix notation into postfix notation.
   (b) Explain briefly various data structures used to implement the symbol table.

7. (a) Explain any four machine independent code optimization techniques.
   (b) Write down the applications of DAG.

8. Explain in detail about machine dependent code optimization techniques with their drawbacks.

*****

Code No: R31051 **R10** **Set No: 4**

III B.Tech. I Semester Supplementary Examinations, June/July - 2014

COMPILER DESIGN

(Computer Science and Engineering)

Time: 3 Hours **Max Marks: 75**

Answer any FIVE Questions

All Questions carry equal marks

*****

1. (a) Explain the functionality of compiler, assembler, interpreter, linker and loader.
   (b) Compare & contrast a pass with a phase with examples.

2. (a) Construct a Finite Automata & Scanning algorithm for recognizing identifiers, numerical constants in C language.
   (b) What are lexical errors? Explain the error-recovery actions taken on the lexical errors.

3. (a) Discuss about the general strategies that a parser can employ to recover from a syntactic error.
   (b) Find FIRSTS & FOLLOWS for the following grammar also construct the LL(1) parsing table:

   $S \to L=R \mid R$
   $L \to *R \mid id$
   $R \to L$

4. (a) Compute LR(0) items for the following grammar:

   $S \to AB$
   $A \to a \mid \varepsilon$
   $B \to b \mid \varepsilon$

   (b) Differentiate between LR parsers and LL parsers.

5. (a) Construct the C LR parsing table for the "dangling-else" grammar.
   (b) Explain Error Recover in LR Parsing.

6. (a) Construct the Syntax Directed Translation scheme to convert a given arithmetic expression into three address code.
   (b) Explain the storage allocation scheme for a block structured language.

7. (a) Explain the principle sources of code-improving transformations.
   (b) Write an algorithm for constructing a basic block.

8. Explain about the following machine dependent code optimization techniques:
   (a) Flow-of-control optimization.
   (b) unreachable-code elimination.
   (c) algebraic simplification.
   (d) peephole optimization.

*****