Roll No. ..... Total Pages : 03

BT-6/M-22 46165

COMPILER DESIGN

PC-CS-302A

Time : Three Hours] [Maximum Marks : 75

Note : Attempt *Five* questions in all, selecting at least *one* question from each Unit. All questions carry equal marks.

Unit I

1. What necessary steps are followed by the compiler for carrying out the compilation process ? Describe in brief the purpose of each step along with a description of the compiler construction tools used in different steps of compilation.

2. What are the rules and properties of Regular Expressions ? What is the relationship between regular expression, lexial analysis, and finite automata ? What does the regular expression $(a + b)^* abb$ mean ? Construct NFA for this regular expression.

Unit II

3. Answer the following questions in brief : 7.

   (a) What is the role of a parser in compilation process ?

   (b) What does ambiguity in a context free grammar mean ?

   (c) How do you represent an abstract syntax tree ? 8.

4. (a) Why are Recursive Descent and LL parsers known as top-down parsers ? How is left recursion eliminated ?

   (b) What is shift-reduced parsing ? Describe how LR parsers use the shift-reduce technique ?

Unit III

5. (a) Is attribute grammar a special case of context free grammar ? Justify. Explain synthesized attributes using an appropriate example.

   (b) Distinguish between Stack allocation and Heap Allocation.

6. (a) Describe in brief the principle ways of organizing and accessing symbol tables.

   (b) What is the significance of generating intermediate code ? Give examples of any *two* intermediate codes used in compilers.

Unit IV

7. What are the types of code optimization ? Describe the ways in which code optimization can be done. Also give suitable example to show how loop optimization can be carried out ?

8. Give brief answers of the following questions :

   (a) How is global data flow analysis used to perform optimization ?

   (b) How does Runtime environment manage runtime memory requirements ?

   (c) Describe any *two* common error-recovery strategies that can be implemented in a parser to deal with errors in the code.