Reg. No. : | | | | | | | | | | |

Question Paper Code: 36021

B.E. / B.Tech. DEGREE EXAMINATION, NOV 2017

Sixth Semester

Computer Science and Engineering

01UCS601 - PRINCIPLES OF COMPILER DESIGN

(Regulation 2013)

Duration: Three hours Maximum: 100 Marks

Answer ALL Questions

PART A - (10 x 2 = 20 Marks)

1. What are the contents present in a symbol table?
2. Distinguish between compiler and interpreter.
3. What are the possible error recovery actions in lexical analyzer?
4. What is the use of an error handler?
5. What is the significance of look-ahead symbols in LR (1) items? When do they loose their significance?
6. Write the drawbacks of shift-reduce parser.
7. Illustrate why every S-attributed definition is L-attributed.
8. What is annotated parse tree?
9. What is flow graph?
10. What is peephole optimization?

PART - B (5 x 16 = 80 Marks)

11. (a) With a neat sketch, discuss the phases of a compiler. (16)

Or

(b) Why is it necessary to study the theory behind the design of compiler? Discuss in detail the cousins of compilers. (16)

12. (a) Convert the given regular expression (a | b)* abb (a | b)* into NFA using Thompson construction and then convert to minimized DFA. (16)

Or

(b) Design a Lexical analyzer generator. Also write the sample code which includes declaration, translation rules and auxiliary procedures. (16)

13. (a) Construct a predictive parsing table for the following grammar:

S → a | ↑ | (T)

T → T, S | S

Explain the behavior of the parser in the following sentences:

(i) (a, (a, a))

(ii) ((a, a), ↑, (a), a). (16)

Or

(b) Consider the following grammar and construct a SLR parsing table for the same

E-> E+T
*T->T*F*
T->F
F->(E)
F->id (16)

14. (a) Write the syntax directed definition for generating 3-address code for an assignment statement. (16)

Or

(b) Discuss the various storage allocation strategies and their merits and demerits. (16)

15. (a) Explain the common sub expression elimination, copy propagation, and transformation for moving loop invariant computations in detail. (16)

Or

(b) Explain the code optimization techniques using examples. (16)
