# B.E. (Computer Engineering)
## PRINCIPLES OF COMPILER DESIGN
### ( 2008 Course) (Semester - I)

*Time : 3 Hours]*                                      *[Max. Marks : 100*

*Instructions to the candidates:*
1) *Answer three questions from Section I and three questions from Section II.*
2) *Answers to the two sections should be written in separate answer books.*
3) *Neat diagrams must be drawn wherever necessary.*
4) *Figures to the right indicate full marks.*
5) *Assume suitable data, if necessary.*

## SECTION - I

**Q1)** a) Explain various phases of compiler with example. List various errors detected in each phase of compiler. **[8]**

b) Design SLR parsing table for following grammar: **[8]**

S → AS|b

A → SA |a

c) Compare Recursive Descent parser and predictive parser. **[2]**

OR

**Q2)** a) Why Lexical analyzer and parser are two separate phases? How they are combined in single pass? **[4]**

b) Show that following grammar is LR (1) but not LALR. **[8]**

S → Aa | aAc |Bc|bBa

A → d

B → d

*P.T.O.*

c) Test whether following grammar is LL (1)? **[6]**

S → iEtSS'|a

S' → eS |∈

E → b

**Q3)** a) What is the need of type checing and type analysis? **[4]**

b) What are synthesized and inherited attributes? Give proper examples.**[4]**

c) What are advantages of Syntax Directed Translation (SDT)? Explain how intermediate code is generated using Top-down translation scheme. **[8]**

OR

**Q4)** a) What is type casting? Explain implicit and explicit type casting, with example. What changes should be made in semantic analyzer to add type casting. **[8]**

b) Differentiate between L-attributed definition and S-attributed definition.**[4]**

c) What is semantic analysis? Give some examples of errors that are detected during semantic analysis. **[4]**

**Q5)** a) Explain commonly used intermediate code representations. Give one example for each. **[8]**

b) Write a syntax directed translation scheme for "if E then S". **[8]**

Generate code for following statement using the above scheme:

if a<b and c>d then a = b +c.

OR

**Q6)** a) Give syntax directed translation scheme for Assignment statement. **[6]**

b) Explain syntax directed translation scheme for Arrays. Generate quadruples for the following: **[10]**

A [i] [j] = B [i] [j] + C [i] [j]

where A, B and C are arrays of size 10 ×20.

# SECTION - II

**Q7) a)** Explain following storage allocation schemes with proper examples.**[12]**

      i)     Stack storage allocation

      ii)    Static storage allocation

      iii)   Heap storage allocation

**b)** For the following 'C' program, show the details of the activation records, if **[6]**

      i)     Stack allocation is used

      ii)    Heap allocation is used

```
main ( )
{
    int * p;
    p = fun ( );
}
int * fun ( )
{
    int i = 23;
    return & i;
}
```

## OR

**Q8) a)** Explain following parameter passing techniques using suitable examples. call by value, call by reference, call restore, call by name. **[8]**

  **b)** Compare static scope with dynamic scope. Illustrate with suitable example. **[10]**

**Q9) a)** Explain Dynamic programming algorithm for code generation. **[8]**

   **b)** Explain with example: **[8]**

   i) Basic blocks and flow graph

   ii) Peephole optimization

<div align="center">OR</div>

**Q10)a)** What are different issues in code generation? **[6]**

   **b)** Explain simple code generation algorithm. Generate code for following 'C' program. **[10]**

```
main ( )
{
    int i;
    int a [10];
    while (i ≤ 10)
    a [i] = 0;
}
```

**Q11)a)** Discuss the principle sources of code optimization. Give proper examples wherever necessary. **[8]**

   **b)** Explain fundamental data flow properties. **[8]**

<div align="center">OR</div>

**Q12)a)** What is "ud chain"? Explain Gen set and Killset for ud chain. **[6]**

   **b)** What is Global common sub-expression? Write an algorithm for elimination of Global common sub-expression. **[6]**

   **c)** Explain in brief: **[4]**

   i) Reaching definitions

   ii) Live variables

<div align="center">✿  ✿  ✿</div>