Total No. of Questions : 12]

SEAT No. :

P812

[Total No. of Pages : 4

[4659] - 225

B.E. (Computer Engineering) (Semester - I)

PRINCIPLES OF COMPILER DESIGN

(2008 Pattern)

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

Instructions to the candidates :

1) ***Answers to the two sections should be written in separate books.***
2) ***Neat diagrams must be drawn wherever necessary.***
3) ***Figures to the right indicate full marks.***

SECTION - I

Q1) a) Explain the interaction between Lexical Analyzer and Parser in first pass of compiler. **[4]**

b) Generate predictive parsing table for the given grammar and parse the string acbbgf. **[10]**

$S \rightarrow a\ B\ D\ h$

$B \rightarrow c\ C$

$C \rightarrow b\ C / E$

$D \rightarrow E\ F$

$E \rightarrow g / E$

$F \rightarrow f / E$

c) Differentiate between SLR, LR(K), and LALR parsers. **[4]**

OR

Q2) a) Differentiate between top-down and bottom-up parsers. **[4]**

b) Generate SLR parsing table for the given grammar and parse the string $id_1 + id_2 + id_3 * id_4$ **[8]**

$E \rightarrow E + T / T$

$T \rightarrow T * F / F$

$F \rightarrow id$

c) Explain : operator Precedence parser **[6]**

P.T.O.

Q3) a) Differentiate between S-attributed and L-attributed grammar. **[4]**

b) What is syntax tree? Give YACC specification to generate syntax tree for expression a + b * c **[8]**

c) Explain : static & Dynamic checking of types **[4]**

OR

Q4) a) Discuss working of Recursive - Descent parser with suitable example. **[6]**

b) What is type checking? Give various type expressions. **[6]**

c) Explain in brief : syntax directed translation. **[4]**

Q5) a) Explain how boolean expressions are evaluated while generating intermediate code. Explain use of marker non-terminals and backpatching. **[6]**

b) Given code : **[6]**

a = b * c + d

write syntax directed translation scheme to translate above code into postfix notation.

c) Explain : Need for Intermediate code **[4]**

OR

Q6) a) Generate 3-addr code for following statements. specify the translation scheme used. **[8]**

A[i] = B [i] + C

P = A [i]

b) Compare : Quadruple, Triple, Indirect Triple **[6]**

c) Generate Triple representation for following : **[2]**

A = – B * (C + D) / E

SECTION - II

Q7) a) Explain : Source Language issues **[4]**

b)` Explain following :

Call by Value, Call by Name, Call by Reference **[6]**

c) Discuss in detail the interaction of symbol table with various phases of compiler. **[6]**

OR

Q8) a) Explain : Issues related to nested procedures **[4]**

b) Explain run-time management of variable length data. **[6]**

c) Discuss storage organization and allocation strategies. **[6]**

Q9) a)` What is DAG? Explain its use in code generation. Generate DAG : **[8]**

$T_1 = A + B$

$T_2 = C + D$

$T_3 = E - T_2$

$T_4 = T_1 - T_3$

b) What is need for next-use Information? Explain how to compute next-use information. **[6]**

c) Explain the concept of basic block. **[2]**

OR

Q10)a) Explain : Issues in code generation **[6]**

b) Determine cost of following instruction sequence. **[6]**

(Clearly mention your assumptions)

MOV b, R_o

ADD c, R_o

MOV R_o, a

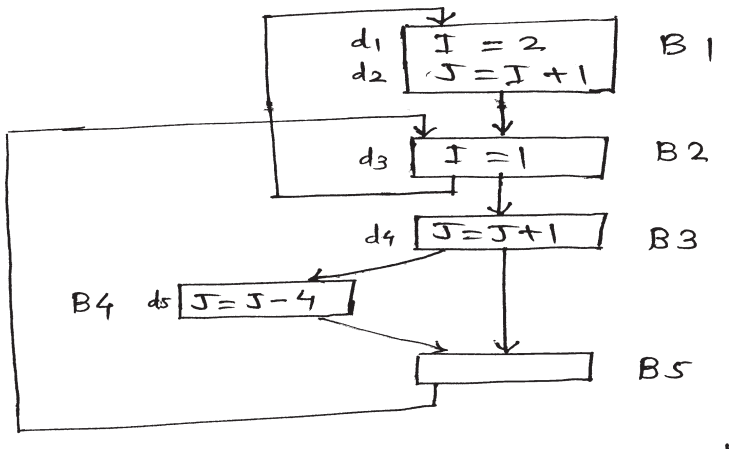
MOV * R_l, * R_o

ADD * R_2, * R_o

c) What is peephole optimization? **[4]**

Q11)a) Which are basic dataflow properties? Explain in detail. **[8]**

b) Given flow graph : **[8]**



Generate IN, OUT, GEN and KILL sets for all blocks.

c) What is control - flow analysis? **[2]**

OR

Q12)a) Explain following optimizations with examples : **[8]**

i) Common Subexpression Elimination

ii) Code movement

iii) Variable Propagation

iv) Strength reduction

b) Explain : Meet Over Path (MOP) solution to data flow problems. **[4]**

c) Explain algorithm for global common subexpression elimination. (Support your answer with example flowgraph). **[6]**