Total No. of Questions : 12]

SEAT No. : □

P701

[Total No. of Pages : 4

[4458]-762

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

PRINCIPLES OF COMPILER DESIGN

(2008 Course)

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.***
4) ***Assume suitable data, if necessary.***

SECTION - I

Q1) a) What is role of Lexical analyzer in compilation process? Explain use and design of input buffer. **[8]**

b) What is LR(k) grammar? Generate LALR parsing table for following grammar : **[10]**

$S \rightarrow Aa|aAc|Bc|bBa$

$A \rightarrow d$

$B \rightarrow d$

OR

Q2) a) Compare : Top - down and Bottom - up parsers. **[4]**

b) What is LL(1) grammar? **[6]**

Compute first and follow sets for following grammar :

$S \rightarrow aAbCD|\in$

$A \rightarrow ASD|\in$

$B \rightarrow SaC|hC|\in$

$C \rightarrow Sf|Cg$

$D \rightarrow aBD|\in$

c) Justify role of stack in the design of Bottom - up parser. Verify whether following grammar is SLR. **[8]**

$S \rightarrow L = R$

$S \rightarrow R$

$L \rightarrow *R$

$L \rightarrow id$

$R \rightarrow L$

Q3) a) What is attribute grammar? Explain role of attribute grammar with SDT. **[6]**

b) Declaration statement in 'C' language is given as :

int a, b, c;

Generate SDT for this statement. **[8]**

c) Why type conversion is required? **[2]**

OR

Q4) a) Given following grammar : **[8]**

$E \rightarrow E + T | T$

$T \rightarrow T * F | F$

$F \rightarrow id$

Write SDT for constructing syntax tree for above grammar, using Top - Down Translation.

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

c) Explain in brief : Recursive descent parser. **[4]**

Q5) a) Explain SDT for following constructs and generate 3 - address code. **[8]**

while a < b and c > d

count = count + 1

b) Compare quadruples and triples. Generate indirect triples for following : **[6]**

$a = b * c - d | - e$

c) What is semantic analysis? **[2]**

OR

Q6) a) Explain SDT for array and generate 3 - addr. code for following : **[8]**

a[i] = b[i, j] + m

With declaration as a[10] and b[20, 20].

b) Explain role of YACC in ICG. (properly explain variables and functions taking part in ICG process.) Also explain how shift - reduce and reduce - reduce conflicts are handled by YACC. **[8]**

SECTION - II

Q7) a) Discuss various data structures for symbol table. **[4]**

b) Explain arrangement of symbol table for block - structured languages. **[4]**

c) Explain : Display. How display is used to access non - local data? **[8]**

OR

Q8) a) Given following program. Show contents of activation record. **[8]**

```
Procedure MAIN ( );
Procedure P(a);
Procedure  Q(b);
L1 : R(x, y);
       end Q;
L2 :  Q(z);
       end P;
       Procedure  R(c,  d);
       end R;
L3 :  P(w);
L4 :  R(u, v);
       end MAIN;
L5 :  MAIN ( );
```

b) Explain : Static scope and Dynamic scope. **[4]**

c) Discuss in brief various parameter passing techniques. **[4]**

Q9) a) Explain simple code Generation algorithm. **[8]**

b) Discuss Register allocation and assignment with respect to code generation. **[8]**

OR

Q10) a) What is next - use information? **[2]**

b) Explain code generator - generator. **[6]**

c) With suitable example, explain dynamic programming algorithm for code generation. **[8]**

Q11) a) Consider following matrix addition program. **[10]**

```
for i = 1 to n
for j = 1 to n
c[i, j] = a[i, j] + b[i, j]
```

i) Generate 3 - addr statements.

ii) Partition the code into basic blocks.

iii) Find loops in the flow graph

iv) Move loop - invariant compution out of the loop.

b) Explain in brief following techniques. **[8]**

i) Constant folding

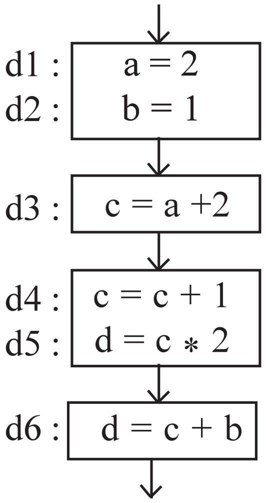
ii) Loop unrolling

iii) Strength reduction

iv) Common subexpression elimination

OR

Q12) a) Consider the flow graph : **[6]**



Compute IN and OUT sets of reaching definitions for each block of flow graph.

b) Explain use of following : **[4]**

i) ud chain

ii) du chain

c) Explain in detail fundamental data flow properties. **[8]**

♦♦♦♦