[This question paper contains 6 printed pages]

6134-A *Your Roll No*

MCA/IV Sem. **J**

MCA 401 – COMPILER DESIGN

(OC)

Time 3 hours *Maximum Marks 60*

(Write your Roll No on the top immediately on receipt of this question paper)

Attempt **all** *questions Parts of a question must be answered together*

1 (a) Describe the language denoted by the following regular expression

$$(0/1) * 0(0/1)(0/1) \quad (2)$$

(b) Write the regular expression for the following language

(i) All strings of 0's and 1's that do not contain the substring 011 (2)

(c) Construct a minimum - state DFA for the following regular expression

$$(a/b) * a(a/b) \quad (5)$$

P.T.O

2 (a) Consider the grammar

bexpr → bexpr or bterm/bterm

bterm → bterm and bfactor/bfactor

bfactor → not bfactor/(bexpr)/true/false

(i) What are terminals, nonterminals and start symbol in the above grammar ? (2)

(ii) Construct the paise tree for the sentence not (true or false). (3)

(iii) Is the above grammar ambiguous ? Why ? (2)

(b) Consider the following grammar

E → E + T/T

T → TF/F

F → F * /a/b

(i) Is the above grammar LL(1) ? Prove or disprove (2)

(ii) Construct the SLR parsing table for this grammar (5)

3 (a) The following grammar generates expressions formed by applying an arithmetic operator + to integer and real constants. When two integers are added, the resulting type is integer, otherwise, it is real

$E \rightarrow E + T/T$

$T \rightarrow num\ num/num$

Give a syntax directed definition to determine the type of each subexpression (5)

(b) Suppose we have following C declaration

```
typedef struct {
        int a, b,
    } node, *head,
node list[100],
head func (int X, node Y)
```

Write type expressions for the types of list and func (2+3)

(c) Translate the following expression into quadraples and indirect triples

$a * (b + c) - (a + b + c) * d$ (2+2)

P.T O

(d) Generate 3-address code for following program –

```
main( )
{
int i,
int a[10],
i = 1,
while (i < = 10) {
   a[i] = 0, i = i+1,
}
}
```

(4)

4 (a) What is printed by the following program assuming (a) Call-by-value (b) Call-by-reference, (c) Copy-restore, (d) Call-by-name

```
Program main(input, output),
    procedure p(x, y, z),
      begin
        y  = y + 1,
        z  = z + x,
      end,
    begin
        a  = 2,
        b  = 3,
        p(a+b, a, a),
        print a,
    end,
```

(4)

(b) Construct a DAG for the following basic block

$$d = b * c$$
$$c = a + b$$
$$b = b * c$$
$$a = c - d \qquad (3)$$

5 (a) What is an activation record ? What all field does it contain ? Which of these field help to access local and global variables ? Explain with example (5)

(b) Consider the following 3-address code Partition it into Basic blocks Create a Flow graph and optimize it by removing common sub-expressions

(1) $i = m - 1$ (2) $j = n$

(3) $t_1 = 4 * n$ (4) $v = a[t_1]$

(5) $i = i + 1$ (6) $t_2 = 4 * i$

(7) $t_3 = a[t_2]$ (8) if $t_3 < v$ goto(5)

(9) $j = j - 1$ (10) $t_4 = 4 * j$

(11) $t_5 = a[t_4]$ (12) if $t_5 > v$ goto(9)

(13) if $i > j$ goto(23) (14) $t_6 = 4 * i$

(15) $x = a[t_6]$ (16) $t_7 = 4 * i$

(17) $t_8 = 4 * j$ (18) $t_9 = a[t_8]$

(19) $a[t_7] = t_9$ (20) $t_{10} = 4 * i$

(21) $a[t_{10}] = x$ (22) goto (5)

(23) $t_{11} = 4 * i$ (24) $x . = a[t_{11}]$

(25) $t_{12} = 4 * i$ (26) $t_{13} = 4 * n$

(27) $t_{14} = a[t_{13}]$ (28) $a[t_{12}] = t_{14}$

(29) $t_{15} = 4 * n$ (30) $a[t_{15}] = x$

(3+2+2)