Total No. of Questions : 10]

SEAT No. :

P2430

[Total No. of Pages :4

[4758] - 601
T.E. (I.T.)
SYSTEMS PROGRAMMING
(2012 Course) (314450) (Semester - II) (End - Semester)

Time : 3 Hours] *[Max. Marks : 70*

Instructions to the candidates:

1) ***Answer Q1 or Q2, Q3 or Q4, Q5 or Q6, Q7 or Q8, Q9 or Q10.***
2) ***Neat diagrams must be drawn wherever necessary.***
3) ***Figures to the right side indicate full marks.***
4) ***Your answers will be valued as whole.***
5) ***Assume suitable data, if necessary.***

Q1) a) For the 'C' code given below, give the different tables that would be generated as output of lexical analysis. **[8]**

```
main ( )
{
    int i, sum, n;
    float avg;
    n=10;
    sum=¢;
    for(i= 1; i<=10 ;i++)
        sum =sum + i;
    avg = sum / (float)n ;
}
```

b) Define macroprocessor and assembler and give examples for each. **[2]**

OR

Q2) a) Give the various data structures in the design of pass-1 of a Two-pass direct linking loader. **[4]**

b) What are the assembler directives? Explain how assembler directives LTORG, ORIGIN and EQU are processed in first pass. **[6]**

P.T.O.

Q3) a) For the following piece of assembly language code, show the contents of symbol table, literal table and pool-tab. Assume size of instruction equal to one. **[5]**

```
START 202
MOVER AREG, ='5
MOVEM AREG, A
LOOP MOVER AREG, A
MOVER CREG, B
ADD CREG, = '1'
MOVEM CREG, B
SUB CREG, A
BC ANY, NEXT
LTORG
ADD CREG,B'
BC LE LOOP
NEXT SUB AREG, = '1'
BC LT, BACK
STOP
ORIGIN 219
MULT CREG, B
A DS 1
BACK EQU LOOP
B DS 1
END
```

b) Define loader and enlist the basic functions of loader. **[5]**

OR

Q4) a) Explain the first three phases of compiler w.r.t. the following statement: **[6]**

r = a – 10 / (c*d^e)

Note: $\wedge$ is exponentiation operator

b) Explain different parameter passing mechanisms in macro-processor. **[4]**

Q5) a) Consider the grammar **[4]**

$E \rightarrow E\text{-}E$

$E \rightarrow E*E$

$E \rightarrow id$

Perform shift Reduce parsing of i/p string "id-id * id"

b) Define table driven predictive parser. For the following grammar. **[10]**

$S \rightarrow AaBb$

$A \rightarrow €$

$B \rightarrow €$

Construct table-driven predictive parser and parse the string "ab".

c) Compare bottom up and top down parser. **[4]**

OR

Q6) a) Consider the following grammar **[10]**

$S \rightarrow (L)|a$

$L \rightarrow L, S|S$

Construct SLR parser and parse for the string (a,(a,a))

b) Explain YACC file structure. **[4]**

c) Explain problem of left factoring in top down parser. **[4]**

Q7) a) Optimize the following code **[8]**

a = x^2

b = 3

c = x

d = c*c

e=b+2

f = a+d

g = e* f

b) Write short note on activation record. **[4]**

c) Explain in brief run time storage allocation. **[4]**

OR

Q8) a) Explain any one technique of machine dependent code optimization. **[6]**

b) Comparison between static, stack & heap allocation. **[6]**

c) Explain machine dependent optimization issues. **[4]**

Q9) a) Design dependency graph for the following grammar. **[4]**

S → T List

T → int

T → float

T → char

T → double

List → List 1/id

b) Translate the following C fragment into the three address code. **[6]**

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

c) Write quadruple and triple for the expression. **[6]**

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

OR

Q10)a) For the following statement, Generate intermediate code in the format: **[8]**

i) Postfix notation

ii) Quadruple

iii) Parse tree

iv) Triple

S = (a+b) / (c–d)

b) Explain concept of type checking. **[4]**

c) Generate three address code for **[4]**

```
while (i<10)
{
    x=0;
    i=i+1;
}
```