Total No. of Questions : 10]

SEAT No. :

P2474

[Total No. of Pages : 4

[5253] -197
T.E. (IT) (Semester - II)
Systems Programming
(2012 Pattern)

Time : 2½ Hours] ***[Max. Marks :70***

Instructions to the candidates:

1) ***Answer Q.1 or Q.2 Q.3 or Q.4, Q.5 or Q.6, Q.7 or Q.8, Q.9 or Q.10***
2) ***Neat diagrams must be drawn wherever necessary.***
3) ***Figures to the right indicate full marks.***
4) ***Assume suitable data, if necessary.***

Q1) a) Define the following terms with examples. **[5]**

i) Compiler

ii) Interpreter

iii) Loader

iv) Assembler

v) Macroprocessor

b) Explain in detail data structures used in Pass I and Pass II of assembler.**[5]**

OR

Q2) a) Generate symbol table, literal table, pool table, Intermediate code and Target Code for the given assembler program. Assume a hypothetical instruction set with every instruction of length 1. **[8]**

```
      START 1000
A     DS  05
      LOAD  A
      SUB AREG, =’10’
      DIV BREG, =’ 10’
      TRANS L
L2    READ TEMP
      LTORG
L     ADD AREG, = ’5’
      SUB BREG, =’ 15’
      ADD  B
```

P.T.O.

```
B     EQU   L+10
      ORIGIN L2 + 20
TEMP DS  5
C     DC 10
      STOP
      END
```

b) List down the phases of a compiler. **[2]**

Q3) a) Compare Compile and Go Loader v/s Absolute Loader. **[4]**

b) For the 'C' code given below, give the different tables that would be generated as output of lexical analysis. **[6]**

```
main ( )
{
    int i = 1, sum = 0, n;
    float avg;
    printf("Enter a value for n:");
    scanf("%d", &n);
    sum = ¢;
    do
    {
        sum = sum + i;
        i ++;
    }
    while (i <= n);
    avg = sum / (float)n;
    printf("average : %f", avg);
    getch ( );
}
```

OR

Q4) a) Write a short note on Lex and Yacc. **[4]**

b) Explain flow chart / Algorithm of pass I of direct linkingloader. **[6]**

Q5) a) With a neat diagram explain the classification of parsers. **[6]**

b) Define table driven predictive parser. For the following grammar

$S \rightarrow aSbS/bSaS/\varepsilon$

Construct table - driven predictive parser and parse the string "ab". **[8]**

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

OR

Q6) a) Using the table given, parse the string id – id* (id-id)/id using an operator precedence parser. **[10]**

	+	-	*	/	^	id	(	)	$
+	>	>	<	<	<	<	<	>	>
-	>	>	<	<	<	<	<	>	>
*	>	>	>	>	<	<	<	>	>
/	>	>	>	>	<	<	<	>	>
^	>	>	>	>	<	<	<	>	>
id	>	>	>	>	>			>	>
(	<	<	<	<	<	<	<	=	
)	>	>	>	>	>			>	>
$	<	<	<	<	<	<	<		

b) Explain the concept of handle in bottom up parsing and explain it w.r.t. the given example : **[4]**

$S \rightarrow SS+/SS*/a$ for the string aaa*a++

c) Compare LALR and CLR parsers. **[4]**

Q7) a) What is SDD? Explain synthesized and inherited attributes with suitable example. **[8]**

b) For the statement given below, generate intermediate code in the form of: **[8]**

i) postfix notation

ii) Parse Tree

iii) Quadruple

iv) Triple

S = (a + b)/(c – d)

OR

Q8) a) Explain concept of type checking. **[4]**

b) For the given piece of code generate TAC : **[8]**

```
X, Y : ARRAY [1 – 10, 1 – 10] OF INTEGER;
for (j = 1; j <=5; j++)
X[2*i – 1][j] = Y[2 * i] [j]
```

c) Explain the need for intermediate code generation. **[4]**

Q9) a) Discuss code generation issues. **[4]**

b) Discuss with suitable example machine dependent code optimization. **[8]**

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

OR

Q10) a) Explain following machine independent optimization techniques : **[8]**

i) Loop invariation

ii) Common sub - expression eimination.

iii) Dead code elimination

iv) Strength reduction

b) Compare machine dependent and independent optimization. **[4]**

c) Explain different storage allocation strategies. **[4]**

♦♦♦♦