

Total No. of Questions : 10]

SEAT No. :

P3641

[5560]-597

[Total No. of Pages : 5

T.E.(I.T.)

SYSTEM PROGRAMMING

(2015 Course) (Semester - II) (314451)

Time : 2½ 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 indicate full marks.
- 4) Assume suitable data if necessary.

Q1) a) For the following piece of assembly Language code. Show the Content of Symbol Table, Literal Table, Pool Table and IC and Machine Code. Assume the machine opcode and size of Instruction as 1. **[6]**

START 200

MOVER AREG, A

L: MOVEM BREG, = '2'

ADD BREG, = '2'

ADD CREG, = '3'

ORIGIN L+20

LTORG

MOVER, AREG, C

C EQU L+15

ADD AREG, = '2'

ADD BREG, = '5'

A DS 5

END

b) Define Loader and Enlist the basic functions of Loader.

[4]

OR

P.T.O.

- Q2) a)** For the following assembly language program show MNT, MDT, ALA and the expanded assembly language program. **[8]**

```
MACRO
XYZ &A
ST 1, &A
MEND
MACRO
MIT &Z
MACRO
&Z &W
AR 4, &W
XYZ ALL
MEND
ST &Z, ALL
MEND
PROG START
USING *, 15
MIT HELLO
ST 2,3
HELLO YALE
YALE EQU 5
ALL DC F '3'
```

END.

- b) How Literals are processed in Assembler. Explain with suitable examples. **[2]**

- Q3) a)** Using the Direct Conversion of RE to DFA algorithm convert the following regular expressions to DFA: $(a + b)^* + (a.c)^* . \#$ **[6]**

- b) Explain following Macro facilities with example. **[4]**

- i) Expansion time Variable
- ii) Change of flow during Macro expansion

OR

- Q4) a)** State True or False and justify your answer : **[5]**
- i) A unit of specification for a program generation through expansion is called as Compiler.
 - ii) An AGO <Sequencing Symbol> statement unconditionally transfers control.
 - iii) APTAB and EVTAB data structures are constructed during pass II of Macro pre-processor.
 - iv) A language processor which bridges an execution gap but is not a language translator is called as detranslator.
 - v) The process of replacement of a character string by another character string during program generation is called as semantic expansion.
- b)** Describe the data structures required in design of two pass Direct Linking Loader with suitable example. **[5]**
- Q5) a)** Differentiate between Top Down and Bottom Up Parser. **[4]**
- b)** Define Handle and Handle Pruning w.r.t Bottom up parser. **[4]**
- For given Grammar $S \rightarrow 0S1 \mid 01$
Identify the handles at each step and Parse the string 000111
- c)** For the following grammar **[8]**
- $$S \rightarrow AaBb$$
- $$A \rightarrow \epsilon$$
- $$B \rightarrow \epsilon$$
- Construct table driven predictive parser and parse the string 'ab'
- OR**
- Q6) a)** Consider the Grammar **[6]**
- $$E \rightarrow E + E$$
- $$E \rightarrow E - E$$
- $$E \rightarrow id$$
- Perform Shift Reduce parsing for given input string "id+id-id"
- b)** Consider the following grammar **[10]**
- $$S \rightarrow (L)a$$
- $$L \rightarrow L, S \mid S$$
- Construct SLR Parser and parse the input string (a,(a,a))

Q7) a) Define and explain annotated parse tree for the given grammar [8]

$L \rightarrow E n$

$E \rightarrow E1 + T$

$E \rightarrow T$

$T \rightarrow T1 * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow \text{DIGIT}$

Annotate the tree for $3*5+4 n$

b) Translate the following C code fragment into three address code (TAC). Assume integer size of 4 bytes; [10]

`int sum = 0,i,j;`

`int A[10][10], B[10][10], C[10][10], X[10];`

`i = 1;`

`j = 1;`

`while (i<10 &&j<=20)`

`{`

`Sum += X[i];`

`C[i][i] = A[i][j]+B[i][j];`

`i++;`

`j++;`

`}`

OR

Q8) a) Design dependency graph for the following grammar [8]

$E \rightarrow E+T / T$

$T \rightarrow T * F / F$

$F \rightarrow \text{id}$

The expression given is: $5+8*10$

b) Translate the following expression [10]

a) $a[i] = b * c - b * d$

b) $x = f(y + 1) + 2$

into Quadruples, Triple, Indirect Triple

- Q9)** a) Write short note on activation record. [4]
- b) Explain following machine independent optimization techniques [8]
- i) Loop in variation.
 - ii) Common sub-expression elimination.
 - iii) Dead code elimination.
 - iv) Strength reduction
- c) Compare machine dependent and independent optimization. [4]

OR

- Q10)** a) Obtain the TAC for the following code before and after applying the optimization techniques using. [12]

- i) Removal of Loop Invariants
- ii) Elimination of common sub expressions

```
int X[10][10], Y[10][10]
```

```
for(i= 1; i<= 10; i++)
```

```
    X[i][2*j - 1] = Y[i][2*j - 1]
```

- b) Explain Code generation issues. [4]

