CENTRAL INSTITUTE OF TECHNOLOGY KOKRAJHAR
(Deemed to be University)
KOKRAJHAR :: BTR :: ASSAM :: 783370

END – SEMESTER EXAMINATION
UG

Session: **Janu-June, 2024** Semester: **VI** Time: **3Hrs.** Full Marks: **100**
Course Code: **UCSE601** Course Title: **Compiler Design**

Answer question no 1 and any *four* from the rest

1. Answer all the questions:
a) Which phase of a compiler group characters into tokens? .....
b) Missing bracket in a *for* loop identified in which phase?
c) Rightmost derivation is related to (top-down / bottom-up parser) ..... ?
d) The most powerful parser is (SLR, Operator Precedence, LALR, CLR)?
e) Which phases of compiler insert data into the Symbol Table?
f) Define Inherited and Synthesized attributes.
g) The mechanism of identifying whether tokens can be generated by a given grammar (Scanner, Parser, DAG, Three address code)?
h) Write the three address code for $a = a + +$
i) Type checking is done during (Scanning,Parsing,SDT,DAG) .....
j) In an LR parser, what do **L** and **R** stand for?

2 x 10

2. (a) Remove left recursion from the following grammar:

$$E \rightarrow E + T|T$$
$$T \rightarrow T * F|F$$
$$F \rightarrow (E)|id$$

(b) Calculate the FIRST and FOLLOW of S in the grammar:

$$S \rightarrow aS|Sb|SS|\epsilon$$

c) Is the grammar in *2.(b)* ambiguous? Justify your answer.

6 + 6 + 8

3. a) Consider the grammar:

$$S \rightarrow iEtS|iEtSeS|a$$
$$E \rightarrow b$$

i) Eliminate left recursion and perform left factoring, if exists.
ii) Create the **LL (1)** parsing table.
b) Is the grammar **LL (1)**? Explain your answer.

$4 + 12 + 4$

4. From the given grammar in *2. (a)*, find the **LR (0)** automata.

20

5. (a) Find the language of the following grammar:

$$S \rightarrow aSbS|bSaS|\epsilon$$

(b) Based on the following SDD draw the annotated parse tree for $(1 + 2)/3 - 2 * 3$

Production	Semantic Rule
expr → expr1 + term	expr.t := expr1.t || term.t || '+'
expr → expr1 - term	expr.t := expr1.t || term.t || '-'
expr → term	expr.t := term.t
term → term1 * factor	term.t := term1.t || factor.t || '*'
term → term1 / factor	term.t := term1.t || factor.t || '/'
term → factor	term.t := factor.t
factor → digit	factor.t := digit.t
factor → (expr)	factor.t := expr.t
digit → 1	digit.t := '1'
digit → 2	digit.t := '2'
digit → 3	digit.t := '3'

(c) What are the types of conflicts present in Shift Reduce Parsing? Define with example. During conversion from CLR to LALR which type of conflict may be introduced?

$5 + 10 + 5$

6. Write the short note on (any two)
a) Basic Block and flow graph
b) Operator Precedence parsing
c) Intermediate Representation

$10 + 10$

———————————( )———————————