Reg. No.										

MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

(A constituent unit of MAHE, Manipal)

VI SEMESTER B.TECH. COMPUTER SCIENCE AND ENGINEERING

MAKEUP EXAMINATIONS, JUNE 2019

SUBJECT: COMPILER DESIGN (CSE 3201)

REVISED CREDIT SYSTEM
(XX-06-2019)

Time: 3 Hours MAX. MARKS: 50

Instructions to Candidates:

- Answer **ALL** the questions.
- Missing data may be suitably assumed.

1A. Explain the different phases of a compiler. **3M**

1B. Explain the two ways by which we can handle reserved words that look like identifiers. **2M**

1C. Write the transition diagram for identifying unsigned numbers in Pascal. Use retract operation wherever required. Also write the pseudo code for implementing the relational operator. **5M**

2A. Check if the given **Grammar G** is LL (1) by constructing a predictive parse table and show the parsing actions for the input string " *11020301033* " Clearly specify the different steps involved during the construction of parse table. **4M**

S → 0 | 1 S 2 S 3 | 1 A 3
A → S | A S
Grammar G

2B. Consider the following C snippet, translate into three address statements. Construct the flow graph and identify the loops in your flow graph. **3M**

```
sum = 0;
for( j = 1; j < = 10; j++)
sum = sum+a[j]-b[j];
flag = 1;
```

2C. Consider the given grammar S→a | ^ | (R), T → S , T | S, R → T **3M**

a. Compute the canonical set of LR(0) items for the above grammar
b. Build the ACTION/GOTO table
c. Use the ACTION/GOTO table to parse the string "(a,a)$"

3A. Construct an LR (1) Automaton for the given **Grammar A**. Also give the number of states that contain reduce operations. **5M**

S → States
States → State | States; State
State → Variable = Expression
Variable → identifier [Expression] | identifier
Expression → identifier | (Expression)
Grammar A

3B. Explain eliminating ambiguity with an example. Write the rules to compute FIRST and FOLLOW sets. **3M**

3C. Explain LR parsing algorithm. **2M**

4A. Generate three address code for the following C segment: (consider array elements of 8 bytes). Draw the quadruple for the generated three address code. **5M**

```
if(( a<=b) and ((c>=d) or (a!=d)))
    z = x – y / z;
else
    z = z + 1;
```

4B. What are the ways for constructing the nodes of a DAG using value number method? Explain. **2M**

4C. What are the addressing modes used in a simple target machine model during the code generation phase. **3M**

5A. What is the use of Lex? Explain with an example the structure of Lex program. **3M**

5B. Explain the principles for designing calling sequences and the layout of activation records. **2M**

5C. Draw annotated parse tree showing dependency edges for the input string "*float w,x,y,z*" using the **Grammar B** given below to evaluate an expression. Also derive the syntax directed definition for the same. **5M**

D → T L
T → int | float
L → L_1 , id | id
Grammar B

****