Registration No : | | | | | | | | | | |

Total Number of Pages : 03

B. Tech
RCS6C002

6th Semester Regular / Back Examination: 2021-22
COMPILER DESIGN
BRANCH(S): CSE,CST, IT
Time : 3 Hour
Max Marks : 100
Q.Code : J357

Answer Question No.1 (Part-1) which is compulsory, any eight from Part-II and any two from Part-III.
The figures in the right hand margin indicate marks.

Part-I

Q1 **Answer the following questions :** (2 x 10)

a) Define lexeme, token and pattern.
b) What do you mean by Handle pruning?
c) When a grammar is said to be in LL(1)
d) What is parser conflict? Mention the type of conflicts that can arise in LR parsers.
e) What is back patching?
f) Mention the advantages of intermediate code generation.
g) What are the objectives of code optimization?
h) What is code motion? Give an example
i) What is an activation record for procedure?
j) Differentiate synthesized and Inherited attributed grammar

Part-II

Q2 **Only Focused-Short Answer Type Questions- (Answer Any Eight out of Twelve)** (6 × 8)

a) Explain the process of compilation. Illustrate the output of each phase of compilation for the input "a = (b+c)* (b+c) * 2"

b) Construct a DFA equivalent to the NFA. Where M=({p,q,r},{0,1}, δ,p,{q,s}) and δ is defined in the following table.

	0	1
→p	{q,s}	{q}
*q	{r}	{q,r}
r	{s}	{p}
*s	-	{p}

c) Write an algorithm for converting a NFA into DFA? Find the NFA for the regular

expression (a+b)*abb. Convert the obtained NFA into DFA.

d) Find the LR(1) items for the following grammar

$S \rightarrow AA$

$A \rightarrow aA \mid b$

e) What are the various ways of calling procedures? Explain in detail.

f) Using Backpatching, generate an intermediate code for following expression

A < B OR C < D AND P < Q

g) What is a three-address code? Mention its types. How would you implement the tree-address statements? Explain with examples.

h) Explain the issues in the design of the code generator.

i) Construct the DAG for the following basic block:

d: = b * c
e: = a + b
b: = b * c
a: = e - d

j) Describe the method of generating syntax-directed definition for control statements.

k) Describe in detail the synax-directed translation of case statement

l) Differentiate various allocation strategies in memory management

Part-III

Only Long Answer Type Questions (Answer Any Two out of Four)

Q3 What are the various issues associated with grammars in Top-down parsing? Identify and resolve the issue(s) if any for the following grammar. **(16)**

$$E \rightarrow E+T \mid T$$

$$T \rightarrow T*F \mid F$$

$$F \rightarrow (E) \mid id$$

Find the predictive parsing table for the given grammar. Show acceptance of the input string " id+id*id" using the constructed predictive parsing table.

Q4 a) List the commonly used intermediate representations. Write the following expression in all types of intermediate representations you know **(8)**

(a – b) * (c + d) – (a + b)

b) Generate three address code for the following program **(8)**

```
while (A < C and B > D) do
        if (A == 3) then C = C + 1
        else
        while ( A <= D) do
                A = A + 3
```

Q5 a) Explain the simple code generator with a suitable example **(8)**

b) Write detailed notes on basic blocks and flow graphs **(8)**

Q6 Draw a snapshot of the execution of the following program showing an activation record with static and dynamic links and other relevant details when 'main' calls 'gun' which in turn, calls 'fun' and 'fun' again calls 'gun' (16)

```
int x=2;
void fun (int n) {
        static int x=1;
        gun(n);
   x--;
}
void gun (int m) {
  int y=m-1;
  if (y>0)
  {
    fun(y);
    x--;
  }
}
main () {gun (x); return (0);}
```