53 (CS 601) CPDG

2018

COMPILER DESIGN

Paper : CS 601

Full Marks : 100

Time : Three hours

The figures in the margin indicate full marks for the questions.

Answer question no. **1** *and* ***any three*** *from the rest.*

1. Consider the following grammar :
$A \rightarrow AB \mid B$, $B \rightarrow BC \mid C$, $C \rightarrow dC \mid e$.

   (a) Create the LR (1) items.

   (b) Using the LR (1) items, create a canonical LR (1) parsing table.

   (c) Check whether "ddde" will be accepted by the parser or not.

(d) Design a LALR(1) parser, equivalent with the canonical LR(1) parser, you have designed in question 1 *(c)*.

15+15+5+5

2. (a) What is left recursion ? Discuss with an example.

(b) Why left recursion elimination is important in top-down parsing ?

(c) What is left factoring ?

(d) Do you believe left factoring can create problem for top-down parsing ? Discuss with an example.

5+5+5+5

3. Consider the following grammar —

$A \rightarrow ABCD \mid BCD \mid CD \mid \in$

$B \rightarrow BCD \mid CD \mid \in$

$C \rightarrow CD \mid D \mid \in$

$D \rightarrow d$

(a) Compute the FIRST and FOLLOW of the above grammar.

(b) State whether the grammar is ambiguous or not. Justify your claim.

15+5

4. Create a basic building block for the following algorithm after generating the intermediate code. 20

```
void func ( )
{
   int i, j ;
   int v, x ;
   if (n<=m) return ;
   i = m–1 ; j = n ; v = a[n] ;
   while (1) {
     do i = i+1 ; while (a[i]<v) ;
     do j = j–1 ; while (a[j]>v) ;
     if (i>=j) break ;
     X = a[i] ; a[i] = a[j] ; a[j] = X ;
     }
     X = a[i] ; a[i] = a[ṅ] ; a[n] = X ;
}
```

5. (a) Create an NFA for the regular expression $(a \cdot b)^* a(a \mid b)^*$ using MYT algorithm.

(b) Construct an equivalent DFA for the above NFA.

5+15

6. (a) Create a DAG for the following expression :

$(X+(X+Y)+Z+(X+Y+Z)+(X+Y+(X+Y+Z)))$

(b) Generate the three-address code.

(c) Represent your three-address code in :

(i) Quadruples and

(ii) Triples format.

5+5+10
