Nirma University

Institute of Technology

Semester End Examination (RPR), May - 2017

B. Tech. in Computer Engineering, Semester-VII

IT794 Compiler Construction

Roll / Exam No. _____ Supervisor's initial with date _____

Time: 3 Hours Max. Marks : 100

Instructions:
1. Attempt all questions.
2. Figures to right indicate full marks.
3. Use section-wise separate answer book.
4. Draw neat sketches wherever necessary.
5. Assume necessary data, wherever required and indicate clearly.

SECTION – I

Q-1. Answer the following **[18]**

(A) State your opinion about following statements with proper justification. (6)

1. Minimizing the number of states in DFA minimizes the memory requirements.
2. Left factoring is the process of factoring out the common prefixed of alternates.
3. Two finite state machines are said to be equivalent if they have the same number of states and edges.
4. An advantage of panic mode of error recovery is that it never gets into an infinite loop.
5. Code optimization is a compulsory phase of compiler.
6. Error recovery scheme for LR parsing table is very important.

(B) Explain phases of a compiler in brief and trace the example: (4)
Interest := (Principle * Rate * Period)/100

OR

(B) Error Recovery strategies play an important role in all phases of compiler, justify with suitable example. (4)

(C) Convert the following NFA – ε in to DFA using subset construction. (8)
Starting state is 0 and accepting state is 8.

State	Move(A, ε)	Move(A, a)	Move(A, b)
0	1,7		
1	2,4		
2		3	
3	6		
4			5
5	6		
6	1,7		
7		8	
8			

Q-2. Answer the following **[16]**

(A) What is "dangling-else" grammar? Eliminate the ambiguity from the following "dangling-else" grammar. (4)

stmt → **if** *expr* **then** *stmt*
| **if** *expr* **then** *stmt* **else** *stmt*
| *other*

(B) Check and Justify whether the following grammar is an operator grammar or not. (4)
E → EAE | (E) | - E |id
A → + | - | * | /

(C) Why lexical analysis is a separate phase in compiler? Trace every possible algorithm of input buffering scheme and show the significance of sentinel forms in input buffering technique. (8)

OR

(C) Show that the following grammar is LR(1) but not LALR(1) (8)
S → Aa | bAc | Bc | bBa
A → d
B → d

Q.3 Answer the following **[16]**

(A) Construct predictive parse table for below given grammar. Trace predictive parsing for input string [a+a-ac] using this table. (8)
S → [S X] | a
X → ε | + S Y | Y b
Y → ε | - S X c

(B) Construct the SLR parsing table for following grammar and also perform parsing of *id = id using it. (8)
S → L = R
S → R
L → * R
L → id;
R → L

SECTION – II

Q-4. Answer the following **[18]**

(A) Write syntax directed definition to identify undeclared variables and multiple time declared variables. (6)

(B) Identify synthesized attribute and inherited attributes from following syntax directed definition. Convert inherited attribute to synthesized attribute. (6)

$D \rightarrow T\ L$ { for all **id** ∈ *L*.list : *addtype*(**id**.entry, *T*.type) }
$T \rightarrow$ **int** { *T*.type := 'integer' }
$T \rightarrow$ **real** { *T*.type := 'real' }
$L \rightarrow L_1$, **id** { *L*.list := L_1.list + [**id**] }
$L \rightarrow$ **id** { *L*.list := [**id**] }

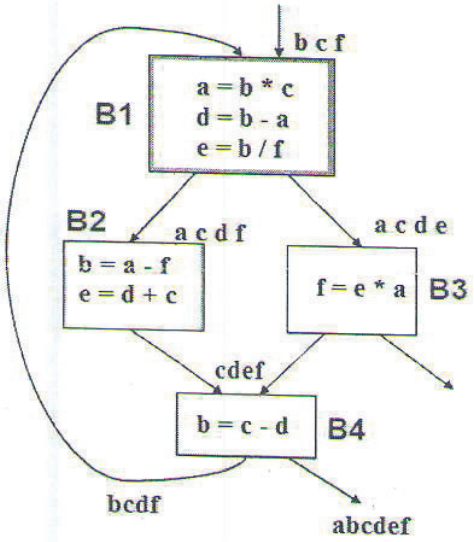
(C) Consider syntax directed definition given in Q.4(B). Generate evaluation order of semantic rules using dependence graph for input string "int a,b,c " (6)

Q-5. Answer the following **[16]**

(A) Eliminate left recursion from following syntax directed definition: (8)

$E \rightarrow E + T$ { E.val = E_1.val + T.val ;}
$E \rightarrow T$ { E.val = T.val ;}
$T \rightarrow T * NUM$ { T.val = T_1.val + NUM.val ;}
$T \rightarrow NUM$ { T.val = NUM.val ;}

(B) Consider following data flow block diagram. Compute usage count for each variable in each block. If system can allocate 3 registers, which variables will be allocated to registers? (8)



OR

(B) Explain graph coloring-interference method to allocate registers for following code fragment: (8)

```
a := read();
b := read();
c := read();
a := a + b + c;
if (a < 10) { d := c + 8;  write(c);}
else if (a < 20) { e := 10;  d := e + a;  write(e);}
else { f := 12;  d := f + a;  write(f); }
write(d);
```

Q-6. Answer the following **[16]**

(A) Apply any three code optimization technique on following code fragment: (8)

```
a=4; b=2; c=3; n=100
a= n-2;
i=0;
j= 4 * i;
while ( i < a)
{
      A[j]=0;
      i=i+1;
      j=j+4;
      b= b *4
}
print(b); print A[];
```

(B) Explain following examples of static type checking with suitable examples: (4)

i) Flow-of-control check
ii) Uniqueness check

OR

(B) Generate code for the following C statements assuming all variables are static and 32 bit integers. (4)

1) x = a – b / d
2) x = a * (e + f)

(C) What is advantage of introducing Marker Nonterminal in translation scheme? Illustrate with appropriate example. (4)