USN : ____________________

Course Code : 18CS62

## Sixth Semester B.E MAKEUP Examination, AUGUST_OCTOBER_2021

## COMPILER DESIGN

Time: 3 hrs

Max. Marks :100

Instructions : Answer any five full questions.

L CO PO M

1a. Explain with a diagram the phases of a Compiler.Show the transition made by each of these phases for the statement a=b+c*5, where a , b and c are reals.

[1] [1] [1, 3] [10]

1b. Design a Lexical Analyzer in C++ to recognize the stream of tokens of C identifiers. Assume the suitable C++ functions to read the character, failure and retract(if necessary) operations.

[3] [1] [3] [10]

2a. What is the need for input Buffereing? Explain the buffer pairs scheme and its drawbacks.

[2] [1] [1] [8]

2b. Design the transition diagram and write the code for the start state, an intermediate state and a final state to recognize the tokens below:
i)Relational operator. ii)unsigned number

[3] [1] [3] [12]

3a. Give the algorithm for the Left Recursion .Analyse the given grammar ,eliminate Left recursion from the following grammar.

A→BC | a
B→CA |Ab
C→AB|CC|a

[3] [2] [3] [10]

3b. state the rules to compute First and Follow.Apply the same algorithm to compute the First and Follow for the given grammar.

E→ E+T | T
T→T*F | F
F→( E ) | id

[2] [2] [1, 3] [10]

4a. Develop predictive parsing table for the following grammar ,show the moves made by the parser for the given input string (a,(a,a))

S →( L ) | a
L →L , S | S

[3] [2] [3] [10]

4b. Develop an Algorithm to left factor a grammar .Give the left factored grammar for the following.

S→iEtS | iEtSeS|a
E→b

[3] [2] [1, 3] [10]

5a. Explain the working of Shift Reduce parser.Illustrate the configuration of a SR for the input using the grammar.

E-->T+E | T
T-->i

[3] [2] [3] [10]

5b. Analyse the given grammar and hence find LR(0) items

S→(S) S

S→€

[3] [2] [3] [10]

6a. Check whether the following grammar is canonical LR by constructing LR(1) collectionfor the given grammar.

S->AaAb | BbBa

A->€

B->€

[2] [2] [3] [10]

6b. Explain the conflicts that occur during shift reduce parsing. Apply and Construct LALR(1) parsing table for the given grammar.

E->( E) | id

[2] [2] [3] [10]

7a. What is Syntax Directed Definition?Write the SDD for a simple desk calculator involving the operators + and *.Show the annotated parse tree for input 7+5*2n

[3] [3] [3] [10]

7b. Explain S attributed SDD , L-attributed SDD and Dependency graph definitions with examples

[2] [3] [2] [10]

8a. Write the SDD for constructing syntax trees for arithmetic expressions consisting of + and - operators.construct syntax tree for the following input:

a+b-c

[3] [3] [3] [10]

8b. Construct L attributed SDD for evaluating arithmetic expressions involving only * operator. construct the dependency graph for the input 2*5

[2] [4] [1] [10]

9a. Construct directed acyclic graph and value number method for the given expression

a + a * (b – c) + (b – c) * d

[3] [4] [3] [10]

9b. List and Explain the various issues in code generation phase

[2] [4] [1] [10]

10a. Apply the code generation algorithm to translate the basic block shown below

t = a- b

u = a – c

v = t + u

a = d

d = v + u

assume t , u , v are temporaries local to the block while a , b , c, d are variables that are line on exit from the block

[3] [4] [3] [10]

10b. Construct three address code and write its Quadruple , Triple and Indirect Triple three address representations

for the following statement

(a+b)*(c+d)-(a+b+c)

[3] [4] [3] [10]

USN : ____________________

Course Code : 16CS63

## Sixth Semester B.E FASTTRACK Examination, AUGUST_SEPTEMBER_2021

## COMPILER DESIGN

Time: 3 hrs

Max. Marks :100

Instructions : Answer any five full Questions.

L CO PO M

1a. Explain with a diagram the phases of a Compiler. Show the transition made by each of these phases for the statement p=q + r * 60, where p , q and r are reals.

[2] [1] [1] [10]

1b. Explain typical Language processing system with a neat diagram.

[2] [1] [1] [6]

1c. Construct a transition diagram to recognize the following tokens.i) Integer constant ii) Identifier.

[2] [1] [1] [4]

2a. What is the need for input buffering ? Explain the buffer pairs scheme and its drawbacks

[2] [2] [2] [10]

2b. Explain with a neat diagram the interaction between Lexical Analyser and the parser.

[2] [2] [1] [6]

2c. What are the applications of a Compiler? Explain.

[2] [2] [1] [4]

3a. Give the algorithm for the Left Recursion .Analyse the given grammar ,eliminate Left recursion from the following grammar.

E→E+T | T<br>
T→T*F |F<br>
F→(E)|id

[3] [2] [2] [10]

3b. state the rules to compute First and Follow.Apply the same to compute First and Follow.

E→ E+T | T<br>
T→T*F | F<br>
F→( E ) | id

[3] [2] [3] [10]

4a. Write an Algorithm to left factor a grammar .Give the left factored grammar for the following.

S→iEtS | iEtSeS | a<br>
E→b

[3] [2] [2] [10]

4b. Develop predictive parsing table for the following grammar ,show the moves made by the parser for the given input string (a,(a,a))

S →( L ) | a<br>
L →L , S | S

[3] [2] [3] [10]

5a. Construct SLR parsing table for the given grammar.

S→AS<br>
A→SA|a

[3] [3] [3] [10]

5b. Construct LALR parsing table for the given grammar.

S'→S<br>
S→CC<br>
C→cC | d

[2] [3] [1] [10]

6a. Analyse the grammar and hence find LR(0) items

S→(S) S

S→€

Explain the working of Shift reduce parser. [4] [2] [3] [10]

6b. Write an algorithm for constructing SLR parsing table and hence Explain the conflicts of Shift reduce parsing with suitable examples [2] [2] [2] [10]

7a. Explain the parser stack implementation of postfix SDT with an example.
Construct directed acyclic graph for the expressiona + a * (b – c) + (b – c) * d [3] [2] [2] [10]

7b. Explain the following with an example
i) Quadruples ii) Triples iii) Indirect Triples [2] [2] [2] [10]

8a. Write syntax directed definition for flow of control statements.
Construct a dependency graph for the declaration float id1 , id2 , id3 [2] [2] [2] [10]

8b. Write annotated parse tree for 6 * 5 + 7m using top down approach . write semantic rules for each. [2] [2] [3] [10]

9a. Apply the code generation algorithm to translate the basic block shown below

t = a- b
u = a – c
v = t + u
a = d
d = v + u

assume t , u , v are temporaries local to the block while a , b , c, d are variables that are line on exit from the block [3] [3] [3] [10]

9b. List and Explain the various issues in code generation phase [2] [3] [2] [10]

10a. What is SDD? Write SDD for a simple desk calculator involving the operators + and *. Show the Annotated parse tree for input 6+7*3n [3] [3] [3] [10]

10b. Construct three address code and write its quadruple , Triple and Indirect Triple three address representations fo the statement (a+b)*(c+d)-(a | b | c) [2] [3] [2] [10]

Scanned by TapScanner

N | | | | | | | | | | |

16CS63/15CS63

**Sixth Semester B.E. Semester End Examination, May/June 2018-19**

# COMPILER DESIGN

me: 3 Hours

Max. Marks: 100

*Instructions:* 1. *Answer five complete questions. Units I and V are compulsory. Answer any one question from each of the remaining units.*
2. *Provide examples, wherever needed.*

## UNIT – I (Compulsory)

L CO PO M

a. Explain with a neat diagram, the phases of a compiler. Show the translation of the input statement: **position = initial * 60** (Assume the variables position and initial to be floats)

(2) (1) (1) (10)

b. Design a lexical analyzer in C++ to recognize an unsigned number. The design should include regular expression / regular definition, transition diagram and C++ implementation code.

(6) (1) (3) (10)

## UNIT – II

L CO PO M

a. Consider the Context free grammar

S→ SS+ ¦ SS* | a

And the string w= aa+a*

a. Give a leftmost derivation for the string b. Give a rightmost derivation for the string

c. Give a parse tree for the string d. Is the grammar ambiguous or unambiguous ? Justify

e. Describe the language generated by the above grammar

(2) (1) (12) (10)

b. What is Left Factoring and why it is required during top down parsing? Write Algorithm for making the grammar left factored. Left factored the following grammar.

S → iEtS | iEtSeS | a
E→ b

(2) (1) (12) (10)

**OR**

a. Give the rules for constructing FIRST and FOLLOW sets. Compute FIRST and Follow functions for the grammar:

S -> A C B | C b B | B a
A -> d a | B C
B -> g | ϵ
C -> h | ϵ

(3) (2) (2) (10)

b. Write an algorithm to construct a predictive parsing table. Construct the predictive parsing table by making necessary changes to the grammar:

E -> E + T | T
T -> T * F | F
F -> ( E ) | id

(3) (2) (2) (10)

## UNIT – III

L CO PO M

a. Explain the working of a shift reduce parser. Illustrate the configurations of a shift-reduce parser for the input: **id*id** and Using the grammar:

E → E+T | T
T → T*F | F

F → (E) | id

State the conflicts that may arise during shift reduce parsing. (3) (2) (1,3) (0

b. Construct SLR Parsing Table for the following grammar:

S → AA

A → Aa | b

Show the moves made by the parser on input: **bab** (3 ) (2) (3) (1

**OR**

5 a. What is LR(k) class of grammars? Explain the model along with the algorithm used for LR Parsing (2) (2) (1) (1

b. Construct canonical sets of LR(1) items for the following grammar

S → AA

A → Aa | b (6) (2) (3) (1

L CO PO M

## UNIT – IV

6 a. What is a Syntax Directed Definition (SDD)? Write the SDD for a simple desk calculator involvin operators + and *. Show the annotated parse tree for input: **8*3+6n** (3) (3) (1,3) (1

b. Write the SDD for constructing syntax trees for arithmetic expressions consisting of + and operators only. Construct syntax tree for: **a – 4+c** (3) (3) (3) (1

**OR**

7 a. Write L-attributed SDD for evaluating arithmetic expressions involving only * operator. Constru the dependency graph for input:5*6 (3) (3) (3) (1

b. Construct Three-Address Code and Write its Quadruple, Triple and Indirect Triple three-addre representations for the following statement:

**a = b * - c + b * - c** (4) (3) (1,3) (1

L CO PO M

## UNIT –V (Compulsory)

8 a. List and explain in brief, various issues in code generation phase. (2) (4) (1) (1

b. What are basic blocks and flow graphs? Write the algorithm for partitioning three addre instructions into basic blocks. (2) (4) (1) (1

SN □□□□□□□□□□ **16CS63**

## Sixth Semester B.E. Makeup Examination, May/June 2018-19

## COMPILER DESIGN

Time: 3 Hours **Max. Marks: 100**

***Instructions:*** *1. UNIT-I & UNIT-V are compulsory*
*2. Answer any one full question from remaining each UNITS.*

### UNIT – I (Compulsory)

L CO PO M

a. Explain various phases of compiler with a neat diagram.
(2) (1) (12) (10)

b. Write the transition diagram along with program code for recognizing relational operators.
(3) (1) (1) (10)

### UNIT – II

L CO PO M

a. What is a Context Free Grammar? When is a grammar ambiguous? Illustrate ambiguity in the following grammar using the input: **id * id + id**

**E →E + E | E * E | (E) | id**

Construct the unambiguous grammar for the above ambiguous grammar.
(4) (2) (1,3, 12) (10)

b. What grammar transformations are needed to make a grammar suitable for top down Parsing? Explain with relevant example(s) why these transformations are needed. Apply them to make the following grammar suitable for top down parsing:

**S → SS+ | SS* | a**
(3) (2) (1,3) (10)

**OR**

3 a. What is a Recursive Descent Parser (RDP)? Write the procedure for a nonterminal in RDP. Show the trace for input: **cad** using the grammar:

**S →cAd**
**A → ab | a**
(3) (2) (1,3) (10)

b. Check whether the following grammar is LL(1) without constructing the table. Further validate the answer by constructing the predictive parsing table.

**S →AaAb | BbBa**
**A → ε**
**B → ε**
(5) (2) (3) (10)

### UNIT - III

L CO PO M

4 a. Explain the working of shift reduce parser. Explain the conflicts of shift reduce parsing with suitable example.
(2) (2) (12) (10)

b. Construct the canonical LR(1) Item sets for the following grammar:

S -> S A | A
A -> a
(3) (2) (2) (10)

**OR**

5 a. Write a schematic of LR parser. Write the canonical collection of set of LR(0) items for the following grammar:

E -> E + T | T
T -> T * F | F
F -> ( E ) | id
(3) (2) (2) (10)

Note: L (Level), CO (Course Outcome), PO (Programme Outcome), M (Marks)

b. Construct LALR parser for the following grammar using LR(1) items:

S' -> S
S -> C C
C -> e D | d

(4) (2) (2) (1

## UNIT - IV

L CO PO M

6 a. Explain the parser stack implementation of postfix SDT with an example.

(2) (3) (12) (1

b. Develop SDD to produce directed acyclic graph for an expression. Show the steps for constructin the directed acyclic graph for the expression: a + a * (b - c) + (b - c) * d.

(3) (3) (3) (1

## OR

7 a. Give the SDD for simple desk calculator and draw dependency graph for the expression: 1 * 2 * 3 * (4 + 5) n.

(3) (3) (3) (1

b. Explain the common three-address instruction forms.

(2) (3) (12) (1

## UNIT –V (Compulsory)

L CO PO M

8 a. Discuss the various issues in the design of a code generator.

(2) (4) (12) (1

b. Generate intermediate code and identify basic blocks for the following program segment:

```
for i from 1 to 10 do
        for j from 1 to 10 do
                a[i,j] = 0.0;
for i from 1 to 10 do
        a[i,i] = 1.0;
```

(3) (4) (3) (1

N □□□□□□□□□□ **15CS63**

## Sixth Semester B.E. Semester End Examination, May / June 2018

## COMPILER DESIGN

me: 3 Hours Max. Marks: 100

*Instructions:* *1. UNIT IV and V are compulsory.*
*2. Answer any ONE full question from remaining each unit.*
*3. Assume any missing information.*

### UNIT – I

a. Explain the typical language processing system with a neat diagram **08 M**

( Level [2 ], CO [ 1], PO [1] )

b. Illustrate the Compilation of the statement **P = I+ R* 60 (P, I, and R are all float values )** for each phase of the Compilation process. **12 M**

( Level [2], CO [ 1], PO [3] )

**OR**

a. With neat diagram demonstrate the interaction between lexical analyzer and the parser. Explain input buffering scheme. **08 M**

( Level [ 2], CO [1 ], PO [1] )

b. Construct a transition diagram to recognize the relational operators of C language and design a Lexical Analyzer in C++ using techniques suitable for hand implementation to recognize the relational operators. Assume the suitable C++ functions to read the character, failure and retract(if necessary) operations **12 M**

( Level [3, 6 ], CO [1 ], PO [3] )

### UNIT – II

3 a. Construct predictive parsing table for the following grammar after applying necessary transformations. **12 M**

$S \rightarrow (L) \mid a$

$L \rightarrow L, S \mid S$

Show the moves made by the parser on input: **(a, a)**

( Level [6,2], CO [2], PO [3,5] )

b. Illustrate ambiguity in the following grammar: **08 M**

$E \rightarrow E + E \mid E * E \mid (E) \mid id$

Eliminate ambiguity from the above grammar.

( Level [2], CO [2], PO [1,3] )

**OR**

4 a. Explain the model along with the algorithm used for table driven Predictive Parsing. Outline the error-recovery strategies employed by a parser. **12 M**

( Level [2], CO [2], PO [1] )

b. Prove that the following grammar is not LL(1) without constructing the parsing table. **08 M**

$S \rightarrow iEtSS' \mid a$

$S' \rightarrow eS \mid \varepsilon$

$E \rightarrow b$

( Level [3], CO [2], PO [3,5] )

## UNIT – III

**5** a. Explain the model of LR parser with parsing Algorithm. 10 M

( Level [2], CO [2 ], PO [ 1] )

b. How would you define LR (0) item ? Construct canonical sets of LR(0) items and GOTO graph for the following grammar. 10 M

**S→AS | b**
**A→SA | a**

( Level [1, 5 ], CO [2 ], PO [ 1, 3] )

**OR**

**6** a. Define Handle and Handle pruning? Demonstrate the process of identifying the handle using Handle pruning with the grammar **E→ E+E | E*E | (E) | id** on the following input string. 08 M

**w=id*(id + id)**

( Level [1, 3 ], CO [2 ], PO [1, 3 ] )

b. How would you define LR(1) item ? Construct canonical sets of LR(1) items and GOTO graph for the following grammar. 08 M

**S→ CC**
**C→eC | d**

( Level [1, 5 ], CO [ 2], PO [1, 3 ] )

c How is an ambiguous grammar handled by LR parser? 04 M

( Level [1], CO [ 2], PO [1] )

## UNIT – IV

**7** a. What is a Syntax Directed Definition (SDD)? Define Synthesized and Inherited attributes. Write the SDD for a simple desk calculator involving operators + and *. Show the annotated parse tree for input: **10+15*4n** 12 M

( Level [1,2,3], CO [4], PO [1,3] )

b. Construct Three-Address Code and represent the same using Quadruples , Triples and Indirect Triple, for the following statement: 08 M

**a = b * - c + b * - c**

( Level [5,6], CO [4], PO [1,3] )

## UNIT – V

**8** a. Describe the various issues in the design of Code generator. 08 M

( Level [1 ], CO [4 ], PO [1] )

b. Outline Simple code generator algorithm and Demonstrate the instructions generated along with the changes in the **Register and Address descriptor** for the following Basic block consisting of three address code statements. 12 M

**t=a – b**
**u=a – c**
**v=t + u**
**a=d**
**d=v + u**

( Level [2, 4 ], CO [4 ], PO [5, 12 ] )

USN □□□□□□□□□□ **15CS63**

**Sixth Semester B.E. Makeup Examination, June 2018**

# COMPILER DESIGN

**Time: 3 Hours** **Max. Marks: 100**

*Instructions: 1. Units IV and V are compulsory. Answer any one question from each of the remaining units.*
*2. Provide examples, wherever needed.*

## UNIT – I

a. Explain with a neat diagram, the phases of a compiler. Show the translation of the input statement: position = initial + rate * 60 (Assume position, initial and rate to be floats) **14 M**
( Level [2], CO [1], PO [1] )

b. Design a lexical analyzer in C++ for relational operators of 'C' programming language. **06 M**
( Level [6], CO [1], PO [3] )

**OR**

a. Show with a neat diagram, the interaction between a lexer and a parser. **12 M**
Define the terms: token, pattern and lexeme. Identify the tokens generated for the 'C' input: while ( I < 10 )
( Level [1,2,3], CO [1], PO [1,3] )

b. Compare the two schemes for input buffering used by a lexer. **08 M**
( Level [2], CO [1], PO [1] )

## UNIT – II

a. What is Left Recursive grammar and how would you eliminate it? Apply the technique and eliminate left recursion from the following Grammar. **06 M**

S→AS | b
A→SA | a

( Level [1,3 ], CO [2 ], PO [3, 5 ] )

b. Explain the model of predictive parser with parsing Algorithm. **06 M**
( Level [2], CO [2 ], PO [ 1,3] )

c. Design the parsing table for predictive parser for the following grammar and verify the grammar to be LL(1). **08 M**
S→iEtS | iEtSeS | a
E→b
( Level [3, 4 ], CO [2 ], PO [3] )

**OR**

a. Consider the grammar **06 M**
S→ a | ^ | (T)
T→ T, S | S
Show the Leftmost and Rightmost derivation for the following sentence . (a,(a, a))
( Level [2], CO [2 ], PO [3] )

b. Explain Recursive-Descent parsing algorithm and discuss the difficulties to implement Recursive Descent Parser. **06 M**
( Level [2 ], CO [ 2], PO [1] )

c. What is Left Factoring? Explain the algorithm for Left factoring the grammar G. Apply the technique and left factor the following grammar. **08 M**

S → aS | Aa | Bb
A → abB | aB | cdg | cdeB | cdfB
B→ b

( Level [2, 3], CO [2 ], PO [3 ] )

## UNIT – III

5 a. Construct SLR Parsing Table for the following grammar: 12 M

S → SS+ | SS* | a

Show the moves made by the parser on input: aa*a+

( Level [6,2], CO [2], PO [3,5] )

b. Explain the working of a Shift Reduce Parser. Show with suitable examples the conflicts that may occur during shift reduce parsing. 08 M

( Level [2], CO [2], PO [1] )

## OR

6 a. What is an LR(1) item? Construct canonical LR(1) collection of items and the Automaton for the following grammar: 12 M

S → CC
C → aC | d

( Level [1,6], CO [2], PO [1,3,5] )

b. Compare LL and LR Parsing methods. Also Compare the different kinds of LR parsers 08 M

( Level [2], CO [2], PO [1] )

## UNIT – IV

7 a. How would you define Syntax Directed Definition? Construct semantic rules for the following grammar and show the annotated parse for the string 3+5*4n 06 M

L→ En
E→E+T
E→T
T→T*F
T→F
F › (E)
F→ digit

( Level [1, 3 ], CO [3 ], PO [1, 3 ] )

b. What is DAG? Construct a DAG, Three address code, Quadruple and Triple representation for the following expression. 10 M

a + a * (b-c) + (b-c) *d.

( Level [1, 3 ], CO [4 ], PO [ 1, 3] )

c Construct the semantic rules for translation of while statement of C language. 04 M

( Level [3 ], CO [ 3], PO [ 1, 3] )

## UNIT –V

8 a. Explain in brief, various issues in code generation phase. 10 M

( Level [2], CO [4], PO [1] )

b. Explain the following with suitable examples: 10 M

Basic Blocks
Flow graphs

( Level [2], CO [4], PO [1] )

USN □□□□□□□□□□

15CS63

## B.E. Fasttrack Semester Examination, July / August 2018

## COMPILER DESIGN

Time: 3 Hours

Max. Marks: 100

*Instructions:* *1. UNIT IV & V are Compulsory.*
*2. Answer any one full question from remaining each UNITS.*
*3. Assume any missing information*

### UNIT – I

1 a. What are the difference between Compiler and interpreter 04 M

( Level [1], CO [ 1], PO [1 ] )

b. What is meant by Input buffering? Write an algorithm for Look ahead code with sentinels 06 M

( Level [1 ], CO [1 ], PO [1 ] )

c. With neat diagram explain the various phases of Compiler. 10 M

( Level [1 ], CO [ 1], PO [ 1] )

OR

2 a. With suitable example explain the role of LEXICAL ANALYSER in the compilation process and interaction between parser. 08 M

( Level [2 ], CO [1 ], PO [3 ] )

b. Construct Transition diagram for token **IDENTIFIER** of C language and design a lexical analyzer in C++ using techniques suitable for hand implementation to recognize the same. Assume suitable C++ functions to read, failure and retract operations. 12 M

( Level [3, 6 ], CO [ 1], PO [3,5 ] )

### UNIT – II

3 a. What is an Ambiguous Grammar? Verify the following grammar to be ambiguous on the input: **id + id * id.** 08 M

$E \rightarrow E + E \mid E * E \mid E - E \mid E / E \mid ( E ) \mid id$

( Level [2], CO [2 ], PO [3, 5 ] )

b. What is Left Recursion? Give the Algorithm to eliminate left recursion. Apply the Algorithm and eliminate the left recursion for the following grammar. 12 M

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

( Level [3 ], CO [ 2], PO [3, 5 ] )

OR

4 a. Given the grammar 12 M

$S \rightarrow i\,E\,t\,S \mid I\,E\,t\,S\,e\,S$
$E \rightarrow b$

i. What is left factoring? Left factor the above grammar

ii. Define FIRST and FOLLOW symbols and Construct FIRST and FOLLOW sets for above grammar

( Level [3 ], CO [2], PO [ 3, 5] )

b. Explain the model of Predictive Parser with Parsing Algorithm. 08 M

( Level [2 ], CO [2], PO [1] )

## UNIT – III

5 a. What is Handle and Handle Pruning ? Show the working of Shift Reduce Parser for accepting the input string w= **id + id * id$** by considering the following grammar. 10

$E \rightarrow E + T \mid T$

$T \rightarrow T * T \mid F$

$F \rightarrow ( E ) \mid id$

( Level [3 ], CO [2 ], PO [1,3 ] )

b. Explain the working Model of LR parser with Parsing Algorithm 10

( Level [1], CO [2 ], PO [1 )

**OR**

6 a. What is LR(0) item in SLR parser? Construct canonical sets of LR(0) items for the following grammar. 10

$A \rightarrow ( A ) \mid a$

( Level [1, 5 ], CO [2 ], PO [1, 3] )

b. Give the Algorithm to build the collections of sets of valid LR(1) items along with two procedures CLOSURE and GOTO 10

( Level [1 ], CO [2 ], PO [ 1] )

## UNIT – IV

7 a. Define Synthesized Attribute and Give SDD for simple calculator and Draw the Annotated parse tree for expression 3*5 + 4n 12

( Level [ 1, 2], CO [3 ], PO [1, 3 ] )

b. What is DAG? Construct the DAG for the following expression 08

i. a + b + a + b

ii. a + a * ( b – c ) + (b – c) * d

( Level [5, 6 ], CO [ 3], PO [1, 3 ] )

## UNIT –V

8 a. Discuss the following terms 10

i. Basic Blocks

ii. Next USE information

iii. FLOW graph

( Level [1 ], CO [ 4], PO [1 ] )

b. With an example explain the **finding common sub expression** and Dead **Code elimination** in code optimization. 10

( Level [2 ], CO [ 4], PO [3, 5 ] )