MCKV

# MCKV Institute of Engineering

**Paper Code: PC-IT501**

**Paper Name: Compiler Design**

***Time Allotted:*** **3 Hours** ***Full Marks:*** **70**

*The figures in the margin indicate full marks.*
*Candidates are required to give their answers in their own words as far as practicable.*

## Group – A

## (Multiple Choice Type Questions)

1. Choose the correct alternatives for any ***ten*** of the following: 10×1=10

(i) Which of the following does not interrupt a running process?

a) A device b) Timer c) Scheduler d) Power failure

(ii) A grammar is meaningless

a) If terminal set and non-terminal set are not disjoint
b) If left hand side of a production is a single terminal
c) If left hand side of a production has no non terminal
d) All of these

Which of the following is not an intermediate code form?

a) Postfix notation b) Syntax trees c) Three address codes d) Quadruples

(iii) The optimization technique which is typically applied on loops is

a) Removal of invariant computation b) Peephole optimization
c) Constant folding d) All of these

(iv) Whether a given pattern constitutes a token or not depends on the

a) Source language b) Target language c) Compiler d) All of these

(v)The optimization which avoids test at every iteration is

a) Loop unrolling b) Loop jamming c) Constant folding d) None of these

(vi)We can optimize code by

a) Dead code elimination b) Common subprograms
c) Copy intermediate loop d) Loop declaration

(vii) Which of the following is the most powerful parser?

a) SLR b) LALR c) Canonical LR d) Operator precedence

(viii) A parser with the valid prefix property is advantageous because it

$t_1=(0)$ $t_4=(3)$
$t_2=(1)$ $t_5=(4)$
$t_3=(2)$

a) Detects error as soon as possible b) Detects errors as and when they occur
c) Limits the amount of erroneous output passed to the text phase
d) All of these

(ix)A grammar that produces more than one parse tree for some sentence is called
a) Ambiguous b) Unambiguous c) Regular d) None of these

(x)In operator precedence parsing, precedence relations are denoted
a) for all pair of non-terminals b) for all pair of terminals
c) to delimit the handle d) only for a certain pair of terminals

(xi)Recursive descent parsing is an example of
a) top down parsing b) bottom up parsing
c) Predictive parsing d) none of the above

(xii) YACC builds up
a) SLR parsing table b) Canonical LR parsing table
c) LALR parsing table d) none of them

## Group – B

### (Short Answer Type Questions)

Answer any ***three*** of the following 3×5=15

2. i) Differentiate between single pass and multi pass compiler.
[Module 1/CO1/Understand-LOCQ]

ii) Left factorize the following grammar:
E→iEtS | iEtSeS|a
E→b 2+3
[Module 3/CO2/Apply-IOCQ]

3. Translate the following expression into quadruples and triples:
(a+b)*(c+d)+(a+b+c) 5
[Module 7/CO4/Understand-LOCQ]

4. i) Construct DAG for the following basic block:
d=b*c
c=a+b
b=b*c
a=e-d
[Module 8/CO4/Analyze-IOCQ]

ii) Draw the syntax tree for the following arithmetic expression: a*-(b+c/d)
[Module 3/CO2/Analyze-IOCQ] 3+2

5.i) Describe the phases of analysis model of compiler design.
[Module 1/CO1/Understand-LOCQ] 3
ii) Define Regular Expression. [Module 2/CO1/Remember-LOCQ] 2

6. Identify the tokens which are present in the following conditional statement and calculate how many tokens are there:

```
if(x>=5)
y=10;
else
y=11;
```

[Module 2/CO1/Apply-IOCQ] 5

## Group – C

## (Long Answer Type Questions)

Answer any ***three*** of the following 3×15=45

7. Consider the following grammar:

E'→E
E→E+T | T
T→T*F|F
F→ (E)|id

Construct the LR(0) parsing table for this grammar.
[Module 3/CO2/Create-HOCQ] 15

8.i) Differentiate between quadruple, triple and indirect triple.
[Module 7/CO4/Understand-LOCQ] 3

ii) Demonstrate the following terms with the example given below:
a:=b*(c+d/b)-(e*f)
a) Quadruples
b) Triples
c) Indirect triples [Module 7/CO4/Apply-IOCQ] 12

9.i) Consider the following grammar: E→E+T|T*F|T

T→F

F→(E)|y

Show shift reduce parsing of the string: y+y+y*y. [Module 3/CO2/Apply-IOCQ]

ii) Test whether the following grammar is LL(1) or not:

S→1AB|ε

A→1AC|0C

B→0S

C→1 [Module 3/CO2/Evaluate-HOCQ]

iii) Explain with example synthesized and inherited attribute.

[Module 4/CO3/Understand-LOCQ] 5+5+(2.5×2)

10.i) Generate machine code for the following instruction:

X=a/-(b*c)-d

Assume 3 registers are available. [Module 9/CO4/Apply-IOCQ]

ii) a) Define Activation Record. [Module 6/CO4/Remember-LOCQ]

b) Explain clearly the components of Activation Record.

[Module 6/CO4/Understand-LOCQ]

iii) Represent the regular expression for identifier and numeric constant.

[Module 2/CO1/Apply-IOCQ] 5+(2+3)+(2+3)

11.i) Consider the following grammar:

S→CC

C→cC | d

Construct the canonical collection of LR (1) items and draw the state diagram for this grammar. [Module 3/CO2/Create-HOCQ]

ii) Eliminate left recursion from the following grammar:

E → E + T|T

T → T * F|F

F → (E)|id [Module 3/CO2/Analyse-IOCQ]

iii) Define Peephole optimization. [Module 8/CO2/Remember-LOCQ] (5+3)+5+2

# MCKV Institute of Engineering

**Paper Code: PC-IT501**

**Compiler Design**

*e Allotted* *Full Marks: 30*



*Can*

**MCKV Institute of Engineering**

**Paper Code: PC-IT501**

**Compiler Design**

***Time Allotted:*** **3 Hours** ***Full Marks:*** 70 ×1

*The figures in the margin indicate full marks.*
*Candidates are required to give their answers in their own words as far as practicable.*

## Group – A

### (Multiple Choice Type Questions)

1.Choose the correct alternatives for any ***ten*** of the following: 10×1=10

i)A compiler that runs on one machine and produces code for a different machine is called
a) Cross compilation b) 2 pass compilation
c) One pass compilation d) None of these

ii)Grammar of the programming is checked in .....phases of compiler.
a) Semantic Analysis b) Syntax Analysis c) Code Generation d) Code Optimization

iii)Which of the following is used for grouping of characters into tokens?
a) Code Optimizer b) Code Generator c) Lexical Analyzer d) Syntax Analyzer

iv)A bottom up parser generates
a) Rightmost derivation b) Rightmost derivation in reverse
c) Left most derivation d) Left most derivation in reverse

v)YACC builds up
a) SLR parsing table b) LALR parsing table
c) Canonical LR parsing table d) None of these

vi)Peephole optimization is used in
a) Lexical Analysis b) Syntax Analysis c) Semantic Analysis d) Code Optimization

vii) If a grammar is LALR(1) then it is necessarily

a) SLR(1) b) LR(1) c) LL(1) d) None of these

viii)In flow Graph, each node represents

a) A basic block b) A single instruction c) Only a for loop d) Both for and while loops

ix)Semantic analysis is applied to determine

a) The argument types b) The type of intermediate results

c) Both a) and b) c) None of these

x)The reverse polish notation or suffix notation is known as

a) Infix notation b) Prefix notation c) Postfix notation d) None of these

xi)In the LR parsing table, _____is defined for terminal.

a) ACTION b) REDUCE c) GOTO d) None of these

xii)In regular expressions, the operator '*' stands for

a) Addition b) Concatenation c) Iteration d) Selection

## Group – B

### (Short Answer Type Questions)

Answer any ***three*** of the following 3×5=15

2. Generate Annotated Parse Tree for the string "3+2-4" using the grammar— 5

E→E+T|E-T|T

T→0|1|2|.....|9

3. Define code optimization? Differentiate constant propagation and constant folding. 1+4

4. Eliminate the left recursion from the following production: 5

A→BC|a

B→CA|Ab

C→AB|CC|a

5. Draw a Syntax tree and DAG for the expression, A=B*- C+B*- C. 2+3

6. Define Token, Pattern and Lexeme. Differentiate Compiler and Interpreter. 3+2

Collection of token taken string Collection of "

High to low Use whole Program to produce

Low to High Use single Block of Code from

## Group – C

### (Long Answer Type Questions)

Answer any ***three*** of the following 3×15=45

7. Construct the SLR (1) parsing table for the following grammar: 15

E→E+T|T

T→T*F|F

F→ (E)|id

8. a) Explain the terms Quadruples, Triples and Indirect Triples with the statement A=B*-C+B*-C.

b) Differentiate top-down and bottom-up approach of parsing? 4+4+4+

9. a) Define basic block and flow graph.

b) Consider the following code: (2+2) +6+2

I. I=12

II. J=1

III. T1=10*I

IV. T2=T1+J

V. T3=8*T2

VI. T4=T3-88

VII. A[ T4 ]= 0.0

VIII. J=J+1

IX. IF J<=10 GOTO III

X. I=I+1

XI. IF I<=10 GOTO II

XII. I=1

XIII. T5=I-1

XIV. T6=88*T5

XV. A[ T6 ] =1.0

XVI. I=I+1

XVII. IF I<=10 GOTO II

Find out the basic blocks and draw the flow graph for the above code.

c) Explain ambiguity of grammar?

d) Show that the following grammar is ambiguous:

$E \rightarrow E + E \mid E - E \mid id$

10. a) Explain with example synthesized and inherited attribute.

b) Define type checking. Differentiate between Dynamic and Static type checking.

c) Consider the following grammar:

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid id$

Perform Shift-Reduce parsing for the string id+ id*id. 5+5+5

11. Write the short notes on ***any three*** of the following? 5×3=15

a) Symbol Table

b) Activation Record

c) Left recursion

d) Cross Compiler

e) Context free Grammar

# MCKV Institute of Engineering

**Paper Code: PC-IT501**

**Compiler Design**

***Time Allotted: 1* Hour** ***Full Marks: 30***

*The figures in the margin indicate full marks.*
*Candidates are required to give their answers in their own words as far as practicable.*

## Group – A
## (Multiple Choice Type Questions)

1. Choose the correct alternatives for any ***five*** of the following: 5×1

   i. A top down parser generates
   a. left-most derivation
   b. right-most derivation
   c. left-most derivation in reverse
   d. right-most derivation in reverse

   ii. The reverse polish notation or suffix notation is known as
   a. Infix notation
   b. Prefix notation
   c. Postfix notation
   d. None of these

   iii. A grammar that produces more than one parse tree for some sentence is called
   a. Ambiguous
   b. Unambiguous
   c. Regular
   d. None of these

iv. If x is a terminal, then FIRST(X) is

a. Null

b. [x]

c. X

d. None of the above

v. A compiler that runs on one machine and produces code for a different machine is called

a. cross compiler

b. 2 pass compiler

c. 1 pass compiler

d. none of these

vi. _____ is suitable for development environment.

a. Interpreter

b. Compiler

c. Debugger

d. None of the above

## Group – B

### (Short Answer Type Questions)

Answer any *two* of the following

2. Prove that the following grammar is ambiguous. [CO2/Apply/IOCQ]

E→ E+E | E*E | id

3. Eliminate the left recursion from the following production: [CO2/Apply/IOCQ]

A→ BC | a

B→ CA | Ab

C→ AB | CC | a

$A \rightarrow A\alpha / \beta$

$A \rightarrow \beta A'$

$A' \rightarrow \alpha A' / \epsilon$

4. a. Differentiate between single pass and multi pass compiler. [CO1/Understand/IOCQ]

b. Left factorize the following grammar: CO2/Apply/IOCQ]

E→iEtS | iEtSeS | a

E→b

## Group – C

## (Long Answer Type Questions)

Answer any ***one*** of the following 1×15

5. a. Define input buffering? [CO1/Remember/LOCQ] 3

b. How is input buffering is implemented? [CO1/Remember/IOCQ] 4

c. What are the problems that may arise at the time of implementing input buffering? [CO1/Remember/IOCQ] 4

d. What is sentinel? Why is its use? [CO1/Remember/IOCQ] 2+2

6. Construct the predictive parsing table for the following grammar- [CO2/Apply/IOCQ] 15

$E \rightarrow TE_R$

$E_R \rightarrow +TE_R | \varepsilon$

$T \rightarrow FT_R$

$T_R \rightarrow *FT_R | \varepsilon$

$F \rightarrow (E) | id$

# MCKV Institute of Engineering

**Paper Code: PC-IT501**

**Compiler Design**

*Time Allotted: 1* **Hour** *Full Marks: 30*

*The figures in the margin indicate full marks.*

*Candidates are required to give their answers in their own words as far as practicable.*

## Group – A

## (Multiple Choice Type Questions)

1. Choose the correct alternatives for any ***five*** of the following: 5×1

i. Which of the following does not interrupt a running process?

a) A device b) Timer c) Scheduler d) Power failure

ii. A grammar is meaningless

a) If terminal set and non-terminal set are not disjoint

b) If left hand side of a production is a single terminal

c) If left hand side of a production has no non terminal

d) All of these

iii. Whether a given pattern constitutes a token or not depends on the

a) Source language b)Target language

c) Compiler d) All of these

iv. Which of the following is the most powerful parser?

a) SLR b) LALR

c) Canonical LR d) Operator precedence

v. A parser with the valid prefix property is advantageous because it

a) Detects error as soon as possible

b) Detects errors as and when they occur

c) Limits the amount of erroneous output passed to the text phase

d) All of these

vi. A grammar that produces more than one parse tree for some sentence is called

a) Ambiguous

b) Unambiguous

c) Regular

d) None of these

## Group – B

## (Short Answer Type Questions)

Answer any *two* of the following 2×5

2. i) Define Compiler. [Module 1/CO1/Remember-LOCQ] 2+3=5

ii) Differentiate between Compiler and Interpreter.

[Module 1/CO1/Understand-LOCQ]

3. Identify the tokens which are present in the following conditional statement and calculate how many tokens are there: [Module 2/CO1/Apply-IOCQ] 5

```
if(x>=5)
y=10;
else
y=11;
```

4. i) Differentiate between single pass and multi pass compiler. 2+3

[CO1/Understand/IOCQ]

ii) Left factorize the following grammar: CO2/Apply/IOCQ]

E→iEtS|iEtSeS|a

E→b

## Group – C

**(Long Answer Type Questions)**

Answer any *one* of the following 1×15

5. i) Consider the following grammar: [Module 3/CO2/Apply-IOCQ] 5+5+5=15

$E \rightarrow E+T|T*F|T$

$T \rightarrow F$

$F \rightarrow (E)|y$

Show shift reduce parsing of the string: y+y+y*y.

ii) Test whether the following grammar is LL(1) or not:

$S \rightarrow 1AB|\varepsilon$

$A \rightarrow 1AC|0C$

$B \rightarrow 0S$

$C \rightarrow 1$

iii) Eliminate the left recursion from the following grammar: [Module 3/CO2/Analyse-IOCQ]

$E \rightarrow E+T|T$

$T \rightarrow T*F|F$

$F \rightarrow (E)|id$

6. Consider the following grammar: [Module 3/CO2/Create-HOCQ] 5+10=

$S \rightarrow AaAb \mid bSaS \mid \varepsilon$

i. Obtain the FIRST and FOLLOW sets of the above grammar.

ii. Construct the LL(1) predictive parsing table using the above grammar.

# MCKV Institute of Engineering

## Paper Code : Compiler Design

## Paper Name : PC-IT501

*Time Allotted: 1* Hour | *Full Marks: 30*

*The figures in the margin indicate full marks.*
*Candidates are required to give their answers in their own words as far as practicable.*

### Group – A

### (Multiple Choice Type Questions)

1. Choose the correct alternatives for any ***five*** of the following: 5×1=5

i. YACC builds up
   a) SLR parsing table
   b) Canonical LR parsing table
   c) LALR parsing table
   d) none of the above

ii. Any description error can be repaired by
   a) insertion alone
   b) deletion alone
   c) insertion and deletion both
   d) replacement alone

iii. Inherited attribute is a natural choice in
   a) keeping track of variable declaration
   b) checking for the correct use of L values and R values
   c) Both A and B
   d) None of the above

iv. The graph that shows basic blocks and their successor relationship is called
   a) DAG
   b) Flow graph
   c) Control graph
   d) Hamiltonian graph

v. Type checking is normally done during
   a) Lexical analysis
   b) Syntax analysis
   c) Syntax directed translation
   d) Code optimization

vi. Which of the following is used for grouping of characters into tokens
   a) A parser
   b) Code optimizer
   c) Code generator
   d) Scanner

## Group – B

**(Short Answer Type Questions)**

Answer any ***two*** of the following 2×5=10

2. Draw DAG for the following basic block: [Module 8/CO4/Analyze-IOCQ] 5

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

3. Generate Annotated Parse Tree for the string "3+2-4" using the grammar— 5

E→E+T|E-T|T [Module 4/CO2/Apply-IOCQ]

T→0|1|2|.....|9

4. i) Define type checking. [Module 5/CO3/Remember-LOCQ] 2+3=5

ii) Differentiate between Dynamic and Static type checking.

[Module 5/CO3/Understand-LOCQ]

## Group – C

**(Long Answer Type Questions)**

Answer any ***one*** of the following 1×15

5. Consider the following grammar: [Module 3/CO2/Create-HOCQ] 15

E'→E

E→E+T | T

T→T*F|F

F→ (E)|id

Construct the LR(0) parsing table for this grammar.

6. Construct the SLR(1) parsing table for the following grammar:

E→E+T|T [Module 3/CO2/Create-HOCQ]

T→T*F|F

F→ (E)|id

# MCKV Institute of Engineering

**Paper Code: PC-IT501**

## Compiler Design

*Time Allotted: 1* **Hour** *Full Marks: 30*

*The figures in the margin indicate full marks.*
*Candidates are required to give their answers in their own words as far as practicable.*

**Group – A**

**(Multiple Choice Type Questions)**

1. Choose the correct alternatives for any ***five*** of the following: 5×1

i. **Which of the following is the most powerful parsing method?**

a) LL(1)
b) Canonical LR
c) SLR
d) LALR

ii. **If grammar is LALR(1) then it is necessarily**

a) SLR(1)
b) LR(1)
c) LL(1)
d) None of the above

iii. **In regular expressions, the operator '*' stands for**

a) Addition
b) Concatenation
c) Iteration
d) Selection

iv. **Who is responsible for the creation of the symbol table?**

a) Assembler

b) Compiler

c) Interpreter

d) All of the above

v. In which parsing, the parser constructs the parse tree from the start symbol and transforms it into the input symbol?

a) Bottom-up Parsing

b) Top-down parsing

c) Both a) and b)

d) None of the above

vi. A _____ is a rule to identify tokens.

a) Lexeme

b) Pattern

c) Handle

d) None of the above

## Group – B

### (Short Answer Type Questions)

Answer any *two* of the following 2×5

2. Consider the grammar—E→E+E | E*E | id. Configure the stack implementation of a shift reduce parser for the input string: id+id*id 5

3. Define parsing? Differentiate top-down and bottom-up parsing. 5

4. Write short notes on predictive parsing. 5

## Group – C

### (Long Answer Type Questions)

Answer any ***one*** of the following 1×1

5. 5

a) Describe with diagram the role of a parser. 5

b) Define left factoring with an appropriate example. 5

c) Define left recursive grammar. Explain no left recursive grammar can be LL (1). 5

6. Construct the LR(0) parsing table for the following grammar:

E'→E
E→E+T|T
T→T*F|F
F→ (E) |id

# MCKV Institute of Engineering

**Paper Code: PC-IT501**

**Compiler Design**

***Time Allotted: 1* Hour** ***Full Marks: 30***

*The figures in the margin indicate full marks.*

*Candidates are required to give their answers in their own words as far as practicable.*

## Group – A

## (Multiple Choice Type Questions)

5×1

1. Choose the correct alternatives for any ***five*** of the following:

i. A top down parser generates

a. left-most derivation

b. right-most derivation

c. left-most derivation in reverse

d. right-most derivation in reverse

ii. The reverse polish notation or suffix notation is known as

a. Infix notation

b. Prefix notation

c. Postfix notation

d. None of these

iii. In operator precedence parsing, precedence relations are defined

a. for all pairs of non-terminals

b. for all pairs of terminals

c. to delimit the handle

d. only for certain pair of terminals

iv. If x is a terminal, then FIRST(X) is

a. Null

b. [x]

c. X

d. None of the above

v. A compiler that runs on one machine and produces code for a different machine is called

a. cross compiler

b. 2 pass compiler

c. 1 pass compiler

d. none of these

vi. _____is suitable for development environment.

a. Interpreter

b. Compiler

c. Debugger

d. None of the above

## Group – B

## (Short Answer Type Questions)

Answer any *two* of the following

2. Eliminate the left recursion from the following production:

A→BC | a

B→CA | Ab

C→AB | CC | a

3. What is ambiguous grammar? Prove that the following grammar is ambiguous

S→AB

A→ aa|a

B→ab|a

4. What is 'handle'?

Consider the grammar E→E+E | E*E | id.

From this, find the handles of the right sentential forms of reduction of the string id

## Group – C

## (Long Answer Type Questions)

Answer any ***one*** of the following 1×15

5. Construct the predictive parsing table for the following grammar- 15

$E \rightarrow TE_R$

$E_R \rightarrow +TE_R|\varepsilon$

$T \rightarrow FT_R$

$T_R \rightarrow *FT_R|\varepsilon$

$F \rightarrow (E)|id$

6. 3+4+4+2+2

a. What is input buffering?

b. How is input buffering implemented?

c. What are the problems that may arise at the time of implementing input buffering?

d. What is sentinel? Why is its use?