II

Compiler Design

Contents

1. Introduction and Lexical Analysis 61
2. Parsing and Syntax Directed Translation ... 66
3. Intermediate Code Generation and Code Optimization 81
4. Runtime Environment 86

Description Sheet

COMPILER DESIGN

Chapter 1 : Introduction and Lexical Analysis

- Compiler
- Compiler Stages
- Grouping of Phases
  Passes in a Compiler
- Lexical Analyzer
- Tokens, Lexemes and Patterns
- Specification of Tokens

Chapter 2 : Parsing and Syntax Directed Translation

- Introduction
- Top-Down Parsing
- LL(1) Parsing
- Bottom-Up Parsing
- LR Parser
- Simple LR Parser
- Canonical LR Parsing (CLR) and LALR
- Operator Precedence Parsing
- Hierarchy of Grammar Classes
- Syntax-Directed Definition
- Construction of Syntax Trees
- Bottom-up Evaluation of S-Attributed Definitions
- L-Attributed Definitions
- Bottom-up Evaluation of Inherited Attributes
- Intermediate Code Generation
- Dependency Graph Generation using Semantic Rules (SDT)
- Syntax-Directed Translation for Intermediate Code Generation
- Intermediate Code Representation

Chapter 3 : Intermediate Code Generation and Code Optimization

- Introduction
- Storage Organization
- Storage Allocation Strategies
- 3-Address code to Basic blocks
- Data Flow Analysis
- Control Flow Analysis
- Code Improvement

Chapter 4 : Runtime Environment

- Introduction
- Activation Records

■■■■

1 Introduction and Lexical Analysis

Multiple Choice Questions

Q.1 In some phase of compiler:

Input

temp1 := inttoreal (60)

temp2 := id3 * temp1

temp3 := id2 + temp2

id1 := temp3

Output

temp1 := id3 * 60.0

id1 := id2 + temp1

where temp1, temp2, temp3 are temporary storage, id1, id2, id3 are identifier inttoreal is converting int 60 to real number. The above phase is

(a) code optimizer
(b) code generator
(c) intermediate code generator
(d) None of the above

Q.2 Match **List-I** with **List-II** and select the correct answer using the codes given below the lists:

List-I	List-II
A. Compilation	1. Runtime specification
B. Interpretation	2. Exception
C. Macro definition	3. Object specification
D. Loading	4. Abbreviation

Codes:

	A	B	C	D
(a)	2	3	4	1
(b)	3	2	4	1
(c)	4	2	1	3
(d)	3	1	4	2

Q.3 Match **List-I** with **List-II** and select the correct answer using the codes given below the lists:

List-I	List-II
A. Preprocessor	1. Resolving external reference
B. Assembler	2. Loading the program
C. Loader	3. Producing relocatable machine code
D. Linker	4. Allow user to define shorthand for longer construct

Codes:

	A	B	C	D
(a)	4	3	2	1
(b)	3	4	1	2
(c)	4	3	1	2
(d)	4	2	3	1

Q.4 We can implement the symbol table data structure for a typical compiler by using which of the following?

1. Linear list
2. Array
3. Hash table

(a) 2 only (b) 1 and 2 only
(c) 1 and 3 only (d) 3 only

Q.5 The cost of developing a compiler

(a) is proportional to the complexity of the source language
(b) is proportional to the complexity of the architecture of the target machine
(c) is proportional to the flexibility of the available instruction set
(d) All of the above

Q.6 Backend of compiler includes those phases that depends on
(a) Target machine
(b) Source language
(c) Both (a) and (b)
(d) None of these

Q.7 Frontend of compiler does not include the phase
(a) semantic analysis
(b) intermediate code generation
(c) code optimization
(d) none of the above

Q.8 Which of the following phase of compilation process is an optional phase?
(a) lexical analysis
(b) syntax analysis
(c) code optimization
(d) code generation

Q.9 A compiler running on computers with small memory would normally be
(a) a multi-pass compiler
(b) single pass compiler
(c) a compiler with less number of phases
(d) none of the above

Q.10 Consider the rule of the C programming language — "every variable must be declared before its use." In which of the following phases of the complier will an error violating this rule be detected?
(a) Code generation
(b) Lexical analysis
(c) Syntax analysis
(d) Semantic analysis

[DRDO-2008]

Q.11 Which of the following is the name of data structure in a compiler that is responsible for managing information about variables and their attributes?
(a) Abstract syntax tree
(b) Attribute grammar
(c) Symbol table
(d) Semantic stack

[JNUEE-2009]

Q.12 Match the following:

List-I
A. Lexical analysis
B. Parsing
C. Register allocation
D. Expression evaluation

List-II
1. Graph coloring
2. DFA minimization
3. Post-order traversal
4. Production tree

Codes:

	A	B	C	D
(a)	2	3	1	4
(b)	2	1	4	3
(c)	2	4	1	3
(d)	2	3	4	1

Q.13 Which of the following is invalid preprocessor command in C language?
(a) # define (b) # include
(c) # ifdef (d) None of these

Q.14 Find the type of error produced by the following C code.

```
main( )
{     in/*comment t x;
      floa/*comment*/t gate;
}
```

(a) Semantic analyser
(b) Syntax error
(c) Both (a) and (b)
(d) None of these

Q.15 Match the following groups:

List-I
A. Lexical analyzer
B. Syntax analyzer
C. Type checking
D. Intermediate code generation

List-II
1. Checks the structure of the program.
2. Analysis of entire program by reading each character.
3. High level language is translated to simple machine independent language.

4. Checks the consistency requirements in a context of the program.

Codes:

	A	B	C	D
(a)	1	2	4	3
(b)	2	1	4	3
(c)	2	4	3	1
(d)	1	4	3	2

Q.16 Which of the following is not a functionality of C compiler?
(a) Identifying tokens
(b) Identifying syntax errors
(c) Linking
(d) None of these

Q.17 Match the following groups:

List-I
A. Allocation
B. Relocation
C. Loading
D. Linking

List-II
1. Resolve the symbol references.
2. Alters the address of instructions and data.
3. Makes the program ready to execute by keeping the machine code in main memory.
4. Assigns the required memory space for the program.

Codes:

	A	B	C	D
(a)	3	2	1	4
(b)	3	1	2	4
(c)	4	3	2	1
(d)	4	2	3	1

Q.18 The number of tokens in the following C statement
printf ("i = %d, &i = %x", i, &i);
(a) 3 (b) 26
(c) 10 (d) 21

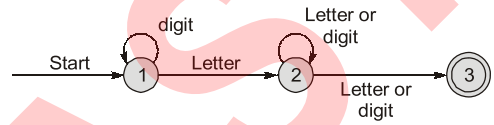
[GATE-2000]

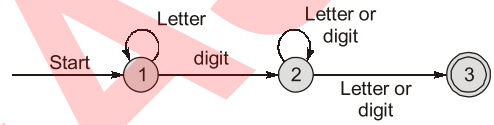
Q.19 Which of the following is a token of C program
(a) # define (b) # include
(c) 123.33 (d) None of these

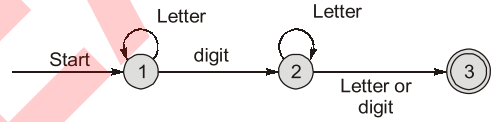
Q.20 Consider the following issues:
1. Simplify the phases
2. Compiler efficiency is improved
3. Compiler works faster
4. Compiler portability is enhanced

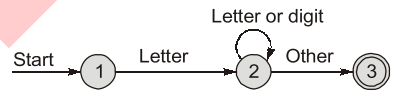
Which is/are true in context of lexical analysis?
(a) 1, 2 and 3 (b) 1, 3 and 4
(c) 1, 2 and 4 (d) All of these

Q.21 Which of the following transition diagram correctly define identifier and keywords?

(a)


(b)


(c)


(d)


Q.22 Consider the following statements:
S_1: The set of string described by a rule is called pattern associated with the token.
S_2: A lexeme is a sequence of character in the source program that is matched by pattern for a token.

Which of the above statements is/are true?
(a) Both S_1 and S_2 are true
(b) S_1 is true S_2 is false
(c) S_2 is true S_1 is false
(d) Both S_1 and S_2 are false

Q.23 If (z == a) function1 (10);
The number of tokens in the above statement are
(a) 9 (b) 11
(c) 10 (d) 12

Q.24 Find the C statement which has a syntax error.
(a) fi (z); (b) for (a, b, c);
(c) whil (a, b); (d) None of these

Q.25 Match the following according to input (from the left column) to the compiler phase (in the right column) that processes it:

List-I

P. Syntax tree
Q. Character stream
R. Intermediate representation
S. Token stream

List-II

(i) Code generator
(ii) Syntax analyzer
(iii) Semantic analyzer
(iv) Lexical analyzer

(a) P → (ii), Q → (iii), R → (iv), S → (i)
(b) P → (ii), Q → (i), R → (iii), S → (iv)
(c) P → (iii), Q → (iv), R → (i), S → (ii)
(d) P → (i), Q → (iv), R → (ii), S → (iii)

[GATE-2017]

Q.26 Consider the following C program:

```
# include <stdio.h>
main ( )
  {
     int x = 10, y = 12;
     char * a;
     a = &x ;
     x = 1xab;
     printf("%d %d", x, * a);
  }
```

Which of the following type of error (earliest phase) is identified during compilation of the program?

(a) Lexical error
(b) Syntax error
(c) Semantic error
(d) None of these

Q.27 A lexical analyzer uses the following patterns to recognize three tokens T_1, T_2 and T_3 over the alphabet {a, b, c}.

T_1 : a? (b | c)*a
T_2 : b? (a | c)*b
T_3 : c? (b | a)*c

Note that 'x?' means 0 or 1 occurrence of the symbol x. Note also that the analyzer outputs the token that matches the longest possible prefix.

If the string bbaacabc is processed by the analyzer, which one of the following is the sequence of tokens it outputs?

(a) $T_1T_2T_3$ (b) $T_1T_1T_3$
(c) $T_2T_1T_3$ (d) T_3T_3

[GATE-2018]

Numerical Answer Type Questions

Q.28 Find the number of tokens in the following C code:

```
main( )
{     int x = 10, *P;
      int y = x ++;
      char *q;
      P = & x; q = 'A';
      if (*P >= 10)
      {     *P = x + 100;
      }
      else
      {     printf("%d", x);
           /*comment */
      }
}
```

Q.29 Consider the following program segment input to the compiler

```
main ( )
{
     int * a, b;
     a = 10;
     a = &b;
     printf("%d%d", b, * a);
     b = */* pointer * /b;
}
```

The number of tokens in the above C code are _____.

Q.30 How many tokens are generated by the lexical analyzer, if the following program has no lexical error?

```
main( )
{
    int x, y;
    fl/*gate */oat z;
    x = /*exam*/10;
    y = 20;
}
```

Q.31 Consider the following program segment:

```
int foo(unsigned int n)
    {
        int c, x = 0;
        While (n! = 0)
          {
            if (n & 01) x ++;
            n >> = 1;
          }
        return c;
    }
```

The number of tokens generated by the Lexical Analyzer in the above program are _____.

Q.32 Find the number of tokens in the following C code using lexical analyzer of compiler.

```
main ( )
{
    /* int a = 10; */
    int * u, * v, s;
    u = &s;
    v = &s;
    printf("%d%d", s, * u);
    // code ended
}
```

Q.33 Consider the following program:

```
1.  main ( )
2.  {     int x = 10;
3.        it (x < 20;
4.        else
5.        y = 20;
6.  }
```

When lexical analyzer scanning the above program, how many lexical errors can be produced?

Multiple Select Questions

Q.34 What are the possible error recovery actions in lexical analysis?

(a) Replacing an incorrect character by a correct character.
(b) Inserting a missing character.
(c) Detecting an extraneous character.
(d) Transposing two adjacent characters.

Q.35 In context of C programming, which of the following is used as a token separator during lexical analysis?

(a) White space
(b) Comment
(c) Semi colon
(d) Equal (=) operator

Try Yourself

T1. Lexical error is _____.

(a) An error produced by lexical analyzer when an illegal character appears.
(b) An error produced by lexical analyzer when a missing left parenthesis in an expression.
(c) An error produced by scanner when both operator and parenthesis appeared consecutively.
(d) All of the above

[Ans: (a)]

■■■■

2 Parsing and Syntax Directed Translation

Multiple Choice Questions

Q.1 Which of the following suffices to convert an arbitrary CFG to an LL(1) grammar?
(a) Removing left recursion alone
(b) Factoring the grammar alone
(c) Removing left recursion and factoring the grammar
(d) None of the above

[GATE 2003]

Q.2 Consider the grammar shown below:

$S \rightarrow i\ E\ t\ S\ S' \mid \alpha$
$S' \rightarrow e\ S \mid \varepsilon$
$E \rightarrow b$

In the predictive parse table. M, of this grammar, the entries M[S′, e] and M[S′, $] respectively are
(a) {S′ → e S} and {S′ → ε}
(b) {S′ → e S} and { }
(c) {S′ → ε} and {S′ → ε}
(d) {S′ → e S, S′ → ε} and {S′ → ε}

[GATE 2003]

Q.3 Consider the grammar shown below.

$S \rightarrow C\ C$
$C \rightarrow c\ C \mid d$

The grammar is
(a) LL (1)
(b) SLR (1) but not LL (1)
(c) LALR (1) but not SLR (1)
(d) LR (1) but not LALR (1)

[GATE 2003]

Q.4 The grammar $A \rightarrow AA \mid (A) \mid \varepsilon$ is not suitable for predictive-parsing because the grammar is
(a) ambiguous (b) Left-recursive
(c) right-recursive (d) an operator-grammar

[GATE 2005]

Q.5 Consider the following grammar:

$S \rightarrow S^*E$
$S \rightarrow E$
$E \rightarrow F + E$
$E \rightarrow F$
$F \rightarrow id$

Consider the following LR(0) items corresponding to the grammar above.
(i) $S \rightarrow S^* .E$
(ii) $E \rightarrow F. + E$
(iii) $E \rightarrow F + .E$

Given the items above, which two of them will appear in the same set in the canonical sets-of-items for the grammar ?
(a) (i) and (ii) (b) (ii) and (iii)
(c) (i) and (iii) (d) None of these

[GATE 2006]

Q.6 Which one of the following is a top-down parser?
(a) Recursive descent parser
(b) Operator precedence parser
(c) An LR(k) parser
(d) An LALR(k) parser

[GATE 2007]

Q.7 Consider the grammar G : $S \rightarrow Aca \mid Bcb$
$A \rightarrow c$
$B \rightarrow c$
where S, A and B are non-terminals and a, b, c are terminal symbols. The G is

(a) LL(2) but not LL(1)
(b) LL(2) but not LR(0)
(c) SLR(1) but not LL(1)
(d) Neither LL(2) nor LR(1)

[DRDO-2009]

Q.8 If a grammar G contains left-recursive productions, then which of the following is NOT TRUE?
(a) A recursive-descent parser for G may loop forever.
(b) G cannot be parsed by any LL(1) parser.
(c) G is ambiguous.
(d) The left-recursive productions can be eliminated by rewriting some productions but some right-recursive productions may be introduced in the resulting grammar.

[DRDO-2009]

Q.9 For a given context-free grammar G
(a) an LR(1) parser can have S-R conflicts if and only if the LR(0) parser has S-R conflicts.
(b) an LALR(1) parser can have reduce-reduce conflicts even if the LR(1) parser doe not have any reduce-reduce conflicts.
(c) an LR(0) parser can have S-R conflicts if the LL(1) parser has S-R conflicts.
(d) an LR(1) parser can have S-R conflicts if the SLR(1) parser S-R conflicts.

[DRDO-2009]

Q.10 Find the best match between the elements of Group-I and Group-II given below:

Group-I	Group-II
P. Data flow analysis	1. Lexical analysis
Q. Regular expression	2. Semantic analysis
R. Type Checking	3. Parsing
S. Pushdown Automata	4. Code Optimization

(a) P-4, Q-1, R-3, S-2 (b) P-2, Q-1, R-4, S-3
(c) P-1, Q-4, R-2, S-3 (d) P-4, Q-1, R-2, S-3

[DRDO-2008]

Q.11 Which of the following is/are true?
I. A left-recursive grammar cannot be LL(1)
II. A right-recursive grammar cannot be LR(1)
III. Every grammar that can be parsed by a canonical LR parser can also be parsed by some SLR parser.
(a) I and II only (b) I only
(c) II only (d) II and III

[DRDO-2008]

Q.12 Consider a state of an LR(0) parser containing the following two items only.

$A \rightarrow abB\bullet$
$C \rightarrow a\bullet b$

Which of the following is false from the information provided above?
(a) There is a shift-reduce conflict in the parsing table
(b) The given grammar is not LR(0)
(c) There is a reduce-reduce conflict in the parsing table
(d) The goto function for this state on symbol b must lead to some state

[DRDO-2008]

Q.13 An LR parser can detect a _____ as soon as it possible when a left to right scan of the input is done.
(a) semantic error (b) syntactic error
(c) logical error (d) lexical error

Q.14 Which of the following statement is correct about parsing?
1. Top-down parsing expands the start symbol to the required string needed whereas in bottom-up parsing reduces the syntax to the start symbol.
2. Top-down parsing is implemented using the set of recursive procedures and implementation of bottom-up parsing is done using the stacks and input buffer.
3. Backtracking is required for both bottom-up and top-down parsing implementation.
(a) 1 and 3 only
(b) 3 only
(c) 1 and 2 only
(d) None of the above

Q.15 The set of prefixes of right sentential forms that can appear on the stack of a shift reduce parser are called ....
(a) right prefixes (b) SR prefixes
(c) viable prefixes (d) non-viable prefixes

Q.16 Which statement is true?
(a) LALR parser is most powerful and costly as compare to other parsers.
(b) All CFG's are LR and not all grammars are uniquely defined.
(c) Every SLR grammar is unambiguous but not every unambiguous grammar is SLR.
(d) LR(K) is the most general back tracking shift reduce parsing method.

Linked Answer for Q.17 & Q.18

Q.17 $S \rightarrow AB$
$S \rightarrow CA$
$B \rightarrow BC$
$B \rightarrow AB$
$A \rightarrow a$
$C \rightarrow aB \mid b$

Check which of the following is correct for above grammar?
(a) the above grammar is in reduce form
(b) the above grammar is not in reduce form
(c) from given non-terminal we get the non-terminal only thus above grammar is in reduce form
(d) None of the above

Q.18 Convert the above grammar to the reduced form if given grammar is not in reduced form
(a) $S \rightarrow CA$
$A \rightarrow a$
$C \rightarrow aB$
(b) $S \rightarrow Ca$
$B \rightarrow AB$
$A \rightarrow a$
$C \rightarrow b$
(c) $S \rightarrow CA$
$A \rightarrow a$
$C \rightarrow b$
(d) Above grammar is already in reduced form

Q.19 Consider the SLR(1) and LALR(1) parsing tables for a context free grammar. Which of the following statements is/are false?
(a) the go to part of both tables may be different
(b) the shift entries are identical in both the tables
(c) the reduce entries in the tables may be different
(d) the error entries in the tables may be different

[GATE-1992]

Q.20 In the following grammar:
$X \rightarrow X \oplus Y \mid Y$
$Y \rightarrow Z * Y \mid Z$
$Z \rightarrow id$
Which of the following is true?
(a) '$\oplus$' is left associative while '*' is right associative
(b) Both '$\oplus$' and '*' are left associative
(c) '$\oplus$' is right associative while '*' is left associative
(d) None of the above

[GATE-1997]

Q.21 Given the following expression grammar:
$E \rightarrow E * F \mid F + E \mid F$
$F \rightarrow F - F \mid id$
Which of the following is true?
(a) * has higher precedence than +
(b) – has higher precedence than *
(c) + and – have same precedence
(d) + has higher precedence than *

[GATE-2000]

Common Data for Q.22 & Q.23

Consider the following expression grammar. The semantic rules for expression calculation are stated next to each grammar production.

$E \rightarrow$ number	E.val = number.val
| E '+' E	$E^{(1)}.val = E^{(2)}.val + E^{(3)}.val$
| E '×' E	$E^{(1)}.val = E^{(2)}.val \times E^{(3)}.val$
;	

[GATE-2005]

Q.22 The above grammar and the semantic rules are fed to a yacc tool (which is an LALR (1) parser generator) for parsing and evaluating arithmetic expressions. Which one of the following is true about the action of yacc for the given grammar?

(a) it detects recursion and eliminates recursion
(b) it detects reduce-reduce conflict, and resolves
(c) it detects shift-reduce conflict, and resolves the conflict in favor of a shift over a reduce action
(d) it detects shift-reduce conflict, and resolves the conflict in favor of a reduce over a shift action

Q.23 Assume the conflicts in Q.22 are resolved and an LALR (1) parser is generated for parsing arithmetic expressions as per the given grammar. Consider an expression $3 \times 2 + 1$. What precedence and associativity properties does the generated parser realize?
(a) equal precedence and left associativity; expression is evaluated to 7
(b) equal precedence and right associativity; expression is evaluated to 9
(c) precedence of '×' is higher than that of '+', and both operators are left associative; expression is evaluated to 7
(d) precedence of '+' is higher than that of '×', and both operators are left associative; expression is evaluated to 9

Q.24 Consider the grammar:

$$S \rightarrow (S) \mid a$$

Let the number of states in SLR (1), LR (1) and LALR(1) parsers for the grammar be n_1, n_2 and n_3 respectively. The following relationship holds good
(a) $n_1 < n_2 < n_3$ (b) $n_1 = n_3 < n_2$
(c) $n_1 = n_2 = n_3$ (d) $n_1 \geq n_3 \geq n_2$

[GATE-2005]

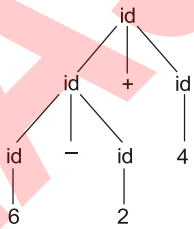
Q.25 Which of the following describes a handle (as applicable to LR-parsing) appropriately?
(a) it is the position in a sentential form where the next shift or reduce operation will occur
(b) it is a non-terminal whose production will be used for reduction in the next step
(c) it is a production that may be used for reduction in a future step along with a position in the sentential form where the next shift or reduce operation will occur.
(d) it is the production p that will be used for reduction in the next step along with a position in the sentential form where the right hand side of the production may be found

[GATE-2008]

Q.26 An LALR(1) parser for a grammar G can have shift-reduce (S-R) conflicts if and only if
(a) the SLR(1) parser for G has S-R conflicts
(b) the LR(1) parser for G has S-R conflicts
(c) the LR(0) parser for G has S-R conflicts
(d) the LALR(1) parser for G has reduce-reduce conflicts

[GATE-2008]

Q.27 Consider the following parse tree.



Which of the following statement is true?
(a) Both + and – are having equal precedence
(b) + is having higher precedence over –
(c) – is having higher precedence over +
(d) None of the above

Q.28 Consider the following productions for a grammar.

$$S \rightarrow i\ E\ t\ S \mid i\ E\ t\ S\ e\ S$$

and configuration of the parser is as follows

Stack	Input
... i E t S	e...

The above production produces which of the following action?
(a) shift e
(b) reduce the configuration
(c) shift/reduce conflict
(d) reduce/reduce conflict

Q.29 Consider the following grammar production.

$$S \rightarrow XX$$
$$X \rightarrow 0X \mid 1$$

In canonical LR method if $[X \rightarrow 0 \cdot X, 0]$ is an item then which of the following is a viable prefix of the item?
(a) 000 (b) 111
(c) 011 (d) 110

Q.30 Consider the LR(0), LR(k), LR(1) and SLR(1) grammars then which of the following about these grammar is true?
(a) LR(0) ⊂ SLR(1) ⊂ LR(1) ⊂ LR(k)
(b) LR(0) ⊂ LR(k) ⊂ LR(1) ⊂ SLR(1)
(c) SLR(1) ⊂ LR(0) ⊂ LR(1) ⊂ LR(K)
(d) LR(K) ⊂ LR(0) ⊂ LR(1) ⊂ SLR(1)

Q.31 For the following grammar which one of the following statement is false?
S → TA
A → ∈ | + TA
T → *i* | *n*
(a) is LL(1)
(b) is LALR(1)
(c) LR(1) but not LALR(1)
(d) is SLR(1)

Q.32 Consider the following grammar G.
1. S → X1
2. X → A1B
3. X → 2
4. A → 2
5. B → A

The above grammar is
(a) Not LL(0), not LL(1), not LR(0), but SLR(1), LR(1) and LALR(1)
(b) Not LL(0), not LL(1), not LR(0), but SLR(1), LR(1) and not LALR(1)
(c) Not LL(0), not LL(1), not LR(0), not SLR(1) but LR(1) and LALR(1)
(d) Not LL(0), not LL(1), not LR(0), not SLR(1), not LR(1) and not LALR(1)

Q.33 A shift reduce parser carries out the actions specified with in braces immediately after reducing with the corresponding rule of grammar
S → AS {print 1}
S → AB {print 2}
A → a {print 3}
B → bC {print 4}
B → dB {print 5}
C → c {print 6}
This syntax directed translation scheme translates a language whose terminal symbols are a, b, c and d into another language whose terminal symbols are 1, 2, 3, 4, 5 and 6 what is the translation of "aaadbc"?
(a) 333546211 (b) 654211
(c) 333645211 (d) 645233311

Q.34 Consider the grammar given below.
A → BA′
A′ → *BA′ | ∈
B → CB′ | ∈
B′ → *CB′ | ∈
C → +Ad | id
The FIRST and FOLLOW of A are respectively
(a) {+, id,*, ∈} and {d, $}
(b) {+, id} and {d, $}
(c) {+} and {d, $}
(d) {id} and {d, *}

Q.35 Consider the following grammar with terminal alphabet Σ = {a, (,), +, *} and the start symbol E. The productions of the grammar are as follows
E → aA | (E)
A → +E | *E | ∈
FIRST (A) and FOLLOW (A) are respectively
(a) {$, ∈ , +, *} and {$,)}
(b) {∈ , +, *} and {$,)}
(c) { +, *,} and {$,)}
(d) {+, *,(} and {$, (}

Q.36 Consider the following grammar.
S → Aa | bAc | dc | bda
A → d
The above grammar is
(a) Not LALR(1) but SLR(1)
(b) Both LALR(1) and SLR(1)
(c) LALR(1) but not SLR(1)
(d) Neither LALR(1) nor SLR(1)

Q.37 What will be the resulting grammar after removal of left-recursion from the following grammar?
E → Ea | Eb | a | b
(a) E → aE′ | bE′; E′ → aE′ | bE′ | ∈
(b) E → aE′ | bE′; E′ → aE | bE | ∈
(c) E → aE′ | bE′ | ∈ ; E′ → aE′ | bE′ | ∈
(d) E → aE′ | bE′; E′ → a | b | ∈

Q.38 Consider the following statements:

S_1: A regular grammar is always linear but not all linear grammar are regular.

S_2: In LL grammar, the usage of productions rule can be predicted exactly, by looking at a llimited part of input.

Which of the above statements is/are correct?

(a) S_1 is true and S_2 is false
(b) S_2 is true and S_1 false
(c) Both S_1 and S_2 are true
(d) Both S_1 and S_2 are false

Q.39 The grammar $S \rightarrow aSa \mid bS \mid c$ is

(a) LL(1) but not LR(1)
(b) LR(1) but not LL(1)
(c) Both LL(1) and LR(1)
(d) Neither LL(1) nor LR(1)

[GATE-2010]

Q.40 Which one of the following is TRUE at any valid state in shift-reduce parsing?

(a) Viable prefixes appear only at the bottom of the stack and not inside
(b) Viable prefixes appear only at the top of the stack and not inside
(c) The stack contains only a set of viable prefixes
(d) The stack never contains viable prefixes

Q.41 A grammar in which all non-empty rules defining the same non-terminal have disjoint selection of first sets, such grammar is called as _____.

(a) LL (1) (b) LR (0)
(c) SLR (1) (d) None of these

Q.42 Consider the following CFG.

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

Which of the following strings not generated by above CFG?

(a) id+id+id
(b) id*id*id
(c) (id*id*id)
(d) None of these

Q.43 Find the ambiguous grammar.

(a) $E \rightarrow E * T \mid T + E \mid T$
$T \rightarrow (T \times id) \mid id$

(b) $E \rightarrow E * T \mid E + E \mid T$
$T \rightarrow (T \times id) \mid id$

(c) Both (a) and (b)
(d) Neither (a) nor (b)

Q.44 Identify an equivalent non-left recursive CFG for the following CFG.

$E \rightarrow E + T \mid E * T \mid T$
$T \rightarrow T - F \mid F$
$F \rightarrow F+2 \mid id$

(a) $E \rightarrow TE'$
$T \rightarrow FT'$
$F \rightarrow idF' \mid id$
$E' \rightarrow *TE' \mid + TE' \mid \varepsilon$
$T' \rightarrow -FT' \mid \varepsilon$
$F' \rightarrow +2F' \mid +2$

(b) $E \rightarrow TE'$
$T \rightarrow FT'$
$F \rightarrow idF' \mid \varepsilon$
$E' \rightarrow *TE' \mid +TE' \mid \varepsilon$
$T' \rightarrow -FT' \mid \varepsilon$
$F' \rightarrow +2F' \mid +2$

(c) $E \rightarrow TE'$
$T \rightarrow FT'$
$F \rightarrow idF'$
$E' \rightarrow *TE' \mid + TE' \mid \varepsilon$
$T' \rightarrow -FT' \mid \varepsilon$
$F' \rightarrow +2$

(d) None of these

Q.45 Find the C statement which has a syntax error.

(a) If (z); (b) For (a, b, c);
(c) While (a, b); (d) All of these

Q.46 Consider the following operator grammar.

$E \rightarrow AaBcD$
$A \rightarrow bA \mid e$
$B \rightarrow bB \mid e$
$D \rightarrow eD \mid g$

Which of the following precedence relation is correct from the above grammar? Assume $x \lessdot y$ is used to represent x is having less precedence than y and in the expression x appears first and then y appears next.

(a) $a \lessdot b$ (b) $c \lessdot g$
(c) $a \lessdot e$ (d) $c \lessdot b$

Q.47 Consider the syntax directed definition shown below.

$S \rightarrow id := E$	{gen (id.place = E.place);}
$E \rightarrow E_1 + E_2$	{t = newtemp ( ); gen ($t = E_1$.place + E_2.place;); E.place = t}
$E \rightarrow id$	{E.place = id.place;}

Here, gen is a function that generates the output code, and newtemp is a function that returns the name of a new temporary variable on every call. Assume that t_i's are the temporary variable names generated by newtemp.

For the statement 'X = Y + Z; the 3-address code sequence generated by this definition is

(a) X = Y + Z
(b) t_1 = Y + Z; X = t_1
(c) t_1 = Y; $t_2 = t_1$ + Z; X = t_2
(d) t_1 = Y; t_2 = Z; $t_3 = t_1 + t_2$; X = t_3

[GATE-2003]

Q.48 Consider the grammar with the following translation rules and E as the start symbol.

$E \rightarrow E_1$ # T	{E, value = E_1.value * T.value}
| T	{E.value = T.value}
$T \rightarrow T_1$ & F	{T.value = T_1.value + F.value}
| F	{T.value = F.value}
$F \rightarrow$ num	{F.value = num.value}

Compute E. value for the root of the parse tree for the expression: 2 # 3 & 5 # 6 & 4.

(a) 200
(b) 180
(c) 160
(d) 40

[GATE-2004]

Q.49 A syntax directed definition specifies

(a) translation of construct in terms of attributes associated with its syntactic component
(b) translation of construct in terms of memory associated with its syntactic component
(c) translation of construct in terms of execution time associated with its syntactic component
(d) None of the above

Q.50 Which of the following attribute can be evaluated by shift reduce parser that executes semantic action only at reduce moves never at shifts?

(a) Synthesized attribute
(b) Inherited attribute
(c) Both (a) and (b)
(d) None of the above

Q.51. Consider the following two statements:

P : Every regular grammar is LL(1).
Q : Every regular set has LR (1) grammar.

Which of the following is TRUE?

(a) Both P and Q are true
(b) P is true and Q is false
(c) P is false and Q is true
(d) Both P and Q are false

[GATE-2007]

Q.52 Which of the following is true?

(a) LR(k) is the most general back-tracking shift-reduces parsing method.
(b) LALR parser is most powerful and costly as compare to other parsers.
(c) Every SLR grammar is unambiguous but not every unambiguous grammar is SLR.
(d) All CFG's are LR and all LR grammar are unambiguous.

Q.53 Consider the following grammar with corresponding synthesized attributes.

$F \rightarrow .L$ {F.v = L.v}

$L \rightarrow L\ B$ {L.len = L_1.len + 1, L.v = L_1.v + $2^{-L.len} \times$ B.v}

$L \rightarrow B$ {L.len = 1, L.v = B.v/2}

$B \rightarrow 0$ {B.v = 0}

$B \rightarrow 1$ {B.v = 1}

If "F.val" gives the value of the binary fraction generated by F in the above grammar then what will be the value of F.val on input .101?

(a) 0.625
(b) 0.550
(c) 0.710
(d) 0.485

Q.54 Which of the following is true?

(a) If grammar is CLR(1) but not LALR(1) then LALR(1) must have reduce-reduce conflict.

(b) If grammar is CLR(1) but not LALR(1) then LALR(1) must have shift-reduce conflict.

(c) If CLR(1) do not have any conflict, then LALR(1) can not have any conflict.

(d) None of the above

Common Data for Q.55 & Q.56

Production rule	Translation rule
$E \rightarrow E \# T$	$\{E.val = E_1.val * T.val\}$ I
I T	{E.val = T.val} II
$T \rightarrow T \& F$	_____.....III
I F	{T.val = F.val} IV
$F \rightarrow num$	{F.val = num} V

Q.55 If the expression 8 # 12 & 4 # 16 & 12 # 4 & 2 is evaluated to 512, then which one of the following correctly represent the rule III?

(a) $T.val = T_1.val * F.val$

(b) $T.val = T_1.val + F.val$

(c) $T.val = T_1.val - F.val$

(d) None of the above

Q.56 The expression 10 # 8 & 6 # 9 & 4 # 5 & 2 will be evaluated to

(a) 300 (b) 12740

(c) 400 (d) None of these

Q.57 A synthesized attribute is an attribute whose value at a parse tree node depends on

(a) attributes at the siblings only

(b) attributes at parent node only

(c) attributes at children nodes only

(d) none of the above

Q.58 An inherited attribute is the one whose initial value at a parse tree node is defined in terms of

(a) attributes at the parent and/or siblings of that node

(b) attributes at children nodes only

(c) attributes at both children nodes and parent and/or siblings of that node

(d) None of the above

Q.59 Match the following errors corresponding to their phase:

Group A

1. Unbalanced parenthesis
2. Appearance of illegal characters
3. Undeclared variables

Group B

A. Syntactic error

B. Semantic error

C. Lexical error

(a) 1 → A, 2 → C, 3 → B

(b) 1 → B, 2 → C, 3 → A

(c) 1 → A, 2 → B, 3 → C

(d) 1 → B, 2 → C, 3 → A

Q.60 Every LALR(1) grammar is

(a) SLR(1) grammar

(b) LL(1) grammar

(c) LR(0) grammar

(d) None of these

Q.61 Operator precedence technique

(a) Simulates a left most derivation

(b) Simulates a right most derivation

(c) Simulates the reverse of right most derivation

(d) Simulates sometimes left most and sometime right most derivation

Q.62 In a bottom-up evaluation of a syntax directed definition, inherited attributes can

(a) always be evaluated

(b) be evaluated only if the definition is L-attributed

(c) be evaluated only if the definition has synthesized attributes

(d) Never be evaluated

[GATE 2003]

Q.63 Consider the translation scheme shown below:

$S \rightarrow T\ R$

$R \rightarrow + T$ {print ('+');} $R \mid \varepsilon$

$T \rightarrow num$ {print (num.val);}

Here num is a token that represents an integer and num.val represents the corresponding integer value. For an input string '9 + 5 + 2', this translation scheme will print

(a) 9 + 5 + 2 (b) 9 5 + 2 +
(c) 9 5 2 + + (d) + + 9 5 2

[GATE 2003]

Q.64 Suppose a program contains infinitely recursive call then which type of error will be generated by the program?
(a) lexical error (b) syntactic error
(c) semantic error (d) logical error

Q.65 Consider the following C fragment
fro (int $x = 0$; $x <= n$; x++)
Which type of error detected by the C compiler for the above code?
(a) lexical error (b) syntactic error
(c) semantic error (d) logical error

Q.66 If conversion from one type to another type is done automatically by the compiler then, it is called
(a) Implicit conversion
(b) Coercions
(c) Both (a) and (b)
(d) None of these

Q.67 Which of the following is not true about dynamic type checking?
(a) Type checking is done during the execution
(b) It increases the cost of execution
(c) All the type error are detected
(d) None of the above

Q.68 The primary sources of semantic error are
(a) Undeclared names
(b) Type incompatibility
(c) Both (a) and (b)
(d) None of the above

Q.69 Consider the following SDT.

$S \rightarrow 2AC$	$\{C.x = A.x\}$
$S \rightarrow 3ABMC$	$\{M.x = A.x + B.x\}$
$C \rightarrow 4$	$\{C.x = 4\}$
$M \rightarrow \varepsilon$	$\{M.x = 0\}$
$B \rightarrow \varepsilon$	$\{B.x = 0\}$
$A \rightarrow \varepsilon$	$\{A.x = 0\}$

Above SDT is
(a) L-attributed grammar only
(b) S-attributed grammar only
(c) Both L-attributed and S-attributed
(d) None of these

Q.70 Every LL(1) grammar is
(a) SLR(1) (b) LALR(1)
(c) LR(1) (d) None of these

Q.71 The type of inadequate states that cannot arise in LR(0) parsing technique
(a) Shift-reduce technique
(b) Reduce-reduce conflict
(c) Shift-shift conflict
(d) None of these

Q.72 $E \rightarrow E + T \mid T$
$T \rightarrow F \uparrow T \mid F$
$F \rightarrow a$
Which of the following is true?
(a) + is left associative and $\uparrow$ is right associative
(b) + is left associative and $\uparrow$ is left associative
(c) + and $\uparrow$ are right associative
(d) None of these

Q.73 $S \rightarrow$ if (E) S
$S \rightarrow$ if (E) Fe | SeS
$F \rightarrow$ for (E; E; E) S
$S \rightarrow a$
$E \rightarrow i$
Above grammar is
(a) Ambiguous
(b) Unambiguous
(c) Left recursive
(d) None of these

Q.74 $S \rightarrow$ while (E) S | a
$E \rightarrow a$
Above grammar is
(a) Not LR(1)
(b) Not SLR(1)
(c) Not LALR(1)
(d) LR(0)

Q.75 The grammar for do-while is
$S \rightarrow$ do S while (E)
$S \rightarrow a$
$E \rightarrow i$
(a) LR(1) but not LALR(1)
(b) LALR(1) but not SLR(1)
(c) SLR(1) but not LR(0)
(d) LR(0) and LL(1)

Q.76 The grammar for nested for statement is
$S \rightarrow$ for (E; E; E) S
$S \rightarrow a$
$E \rightarrow i$
(a) LR(1) but not LALR(1)
(b) LALR(1) but not SLR(1)
(c) LR(1) but not LR(0)
(d) LR(0) and LL(1)

Q.77 Which of the following is true for the given grammar
$E \rightarrow E \uparrow T \mid T - E \mid T$
$T \rightarrow F + E \mid F * T \mid F$
$F \rightarrow a$
(a) $\uparrow$ is right associative
(b) + and – have same precedence
(c) + is higher precedence than *
(d) None of these

Q.78 Consider given SDT:
$E \rightarrow TR$
$R \rightarrow + TMR \mid \in$
$M \rightarrow \in$ {print(+)}
$T \rightarrow$ num {print (num.val)}
If shift/reduce parser carrier out translation what output produce for the input 3 + 3 + 3
(a) 9 (b) 333 + +
(c) 33 + 3 + (d) None of these

Q.79 Consider the SDT:
$E \rightarrow TE'$
$E' \rightarrow *T$ {print(*)} $E' \mid \in$
$T \rightarrow$ id {print(id.name)}
Converting above L-attributed definition to S-attributed definition result in
(a) $E \rightarrow E * T \mid T$ {print(*)}
$T \rightarrow$ id {print(id.name)}
(b) $E \rightarrow TE'$
$E' \rightarrow *TME' \mid \in$
$M \rightarrow \in$ {print(*)}
$T \rightarrow$ id {print(id.name)}
(c) $E \rightarrow T * E$ {print(*)}
$E \rightarrow T$
$T \rightarrow$ id {print(id.name)}
(d) None of these

Q.80 $A \rightarrow B \uparrow A \mid A * B \mid B$
$B \rightarrow B - B \mid C/B \mid C$
$C \rightarrow i$
Given grammar reflects that
(a) $\uparrow$ has higher precedence than –
(b) – has higher precedence than /
(c) – has higher precedence than $\uparrow$
(d) None of these

Q.81 Match the following:

P.	LL(1) is	**1.**	LR(1)
Q.	LR(0) is	**2.**	LL(1)
R.	SLR(1) is	**3.**	LALR(2)
S.	LALR(1) is	**4.**	LR(0)
		5.	SLR(1)
		6.	Operator grammar

Codes:

	P	Q	R	S
(a)	1	5	3	1
(b)	6	4	3	5
(c)	3	5	1	1
(d)	3	2	3	6

Q.82 Consider the SDT given below:
$A \rightarrow aBCD$
{A.i = B.i}
{C.s = a.s}
{B.s = C.s}
The above SDT is
(a) L-attributed
(b) S-attributed
(c) Both (a) and (b)
(d) None of these

Q.83 Consider the grammar given below:

$A \rightarrow SB \mid S$

$B \rightarrow ; SB \mid ;S$

$S \rightarrow a$

Convert into equivalent operator grammar

(a) $A \rightarrow S; A \mid S; S \mid S$
$S \rightarrow a$

(b) $A \rightarrow S; aB \mid S; S \mid S$
$B \rightarrow ; aB \mid ; S$
$S \rightarrow a$

(c) $A \rightarrow S; A \mid S; S \mid S$
$B \rightarrow ; SB \mid ; S$
$S \rightarrow a$

(d) None of these

Q.84 Consider the following LL (1) parsing table

Non Terminals	x	q	r	y	$
S	$S \rightarrow P$	$S \rightarrow P$	$S \rightarrow P$	$S \rightarrow P$	$S \rightarrow P$
P	$P \rightarrow TQR$	$P \rightarrow QR$	$P \rightarrow QR$	$P \rightarrow TQR$	$P \rightarrow QR$
Q		$Q \rightarrow q$	$Q \rightarrow \in$		$Q \rightarrow \in$
R			$R \rightarrow r$		$R \rightarrow \in$
T	$T \rightarrow x$			$T \rightarrow y$	

Which of the following string generates syntax error by the LL (1) parser with start symbol 'S'?

(a) y (b) xqr
(c) xyq (d) All of these

Q.85 Let 'G' be a grammar with the following translations:

$S \rightarrow p\{print\ "G"\}\ P$
$P \rightarrow q\{print\ "A"\}\ Q$
$P \rightarrow r\{print\ "T"\}$
$P \rightarrow \varepsilon\{print\ "E"\}$
$Q \rightarrow s\{print\ "A"\}\ P$
$Q \rightarrow \varepsilon\{print\ "O"\}$

What is the output produced for the input "pqsqsr" using the bottom-up parsing with above translations?

(a) TAAAG (b) TAAAAG
(c) GAAAAT (d) AAAGAT

Q.86 Consider the following two sets of LR(1) items of LR(1) grammar.

$X \rightarrow c.X, c/d$	$X \rightarrow c.X, \$$
$X \rightarrow .cX, c/d$	$X \rightarrow .cX, \$$
$X \rightarrow .d, c/d$	$X \rightarrow .d, \$$

Which of the following statement related to merging of the two sets in the corresponding parser is/are FALSE?

1. Cannot be merged since lookaheads are different.
2. Can be merged but will result in S-R conflict.
3. Can be merged but will result in R-R conflict.
4. Cannot be merged since goto on c will lead to two different sets.

(a) 1 only (b) 2 only
(c) 1 and 4 only (d) 1, 2, 3 and 4

[GATE-2013]

Q.87 Consider the following grammar G.

$S \rightarrow F \mid H$
$F \rightarrow p \mid c$
$H \rightarrow d \mid c$

Where S, F and H are non-terminal symbols, p, d and c are terminal symbols. Which of the following statement(s) is/are correct?

S_1: LL(1) can parse all strings that are generated using grammar G.

S_2: LR(1) can parse all strings that are generated using grammar G.

(a) Only S_1 (b) Only S_2
(c) Both S_1 and S_2 (d) Neither S_1 and S_2

[GATE-2015 (Set-3)]

Q.88 Consider the following Syntax Directed Translation Scheme (SDTS), with non-terminals {S, A} and terminals {a, b}.

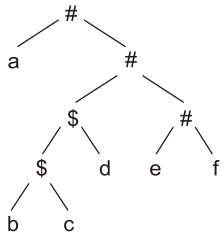
$S \rightarrow aA$	{print 1}
$S \rightarrow a$	{print 2}
$A \rightarrow Sb$	{print 3}

Using the above SDTS, the output printed by a bottom-up parser, for the input aab is:

(a) 1 3 2 (b) 2 2 3
(c) 2 3 1 (d) syntax error

[GATE-2016 (Set-1)]

Q.89 Consider the following parse tree for the expression a # b $ c $ d # e # f, involving two binary operators $ and #.



Which one of the following is correct for the given parse tree?

(a) $ has higher precedence and is left associative; # is right associative

(b) # has higher precedence and is left associative; $ is right associative

(c) $ has higher precedence and is left associative; # is left associative

(d) # has higher precedence and is right associative; $ is left associative

[GATE-2018]

Numerical Answer Type Questions

Q.90 Consider the following augmented grammar G which is used to build LR (0) parsing table.

$E' \rightarrow E$

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid id$

How many rows are present in the parsing table?

Q.91 Consider the following CFG.

$S \rightarrow SA \mid a \mid b$

$A \rightarrow @SB \mid B\$ \mid \varepsilon$

$B \rightarrow \bullet SAg \mid \varepsilon$

How many reductions will happen using bottom-up parser to parse the string: "a @ b $.ag"?

Q.92 Consider the following grammar.

Stmts → Stmt | Stmts; Stmt

Stmt → Var = E

Var → id [E] | id

E → id | (E)

Find the number of conflicts in LR(0).

Q.93 Consider the following grammar.

S → ABA

A → Bc | dA | ε

B → eA

How many entries have multiple productions in LL(1) table?

Q.94 Consider the following CFG.

E → E × T | T + E | T | (E)

T → (T × F) | F

F → id

How many parse trees can be constructed to derive the string "id+(id × id)" from the above CFG?

Q.95 Consider the following CFG.

S → aAb | aBc | bAd | bBe

A → g

B → g

How many states are exist in DFA using LALR (1) constructions for the above grammar?

Q.96 Consider the following CFG with a start symbol G.

$G \rightarrow ATE \mid g \mid \varepsilon$

$A \rightarrow TE \mid a \mid \varepsilon$

$T \rightarrow t \mid \varepsilon$

$E \rightarrow e \mid \varepsilon$

How many derivation trees exist to derive the string "te"?

Multiple Select Questions

Q.97 Consider the following grammar G:

$S \rightarrow bAd \mid bBe$

$A \rightarrow a$

$B \rightarrow a$

The above grammar G is

(a) LR(0) (b) SLR(1)

(c) LR(1) (d) LL(1)

Q.98 Which of the following are correct?

(a) LR parsers can be constructed to recognize most of the programming languages for which the context free grammar can be written.

(b) The class of grammar that can be parsed by LR parser is a superset of class of grammars that can be parsed using predictive parsers.

(c) LR parsers work using non backtracking shift reduce technique.

(d) LL(1) is type of LR parsers.

Q.99 What are the demerits of LALR parser?

(a) Merger in LALR parser may produce shift / reduce conflict.

(b) After the LR parser has declared an error, LALR parser never shift a symbol after the LR parser declared an error.

(c) Merger in LALR parser may produce reduce/ reduce conflict.

(d) On erroneous input, LALR parser may proceed to do some reductions.

100. Which of the following are the reasons for being LR parsers more effective than other parsers?

(a) LR parsers can be constructed for all programming language constructs for which CFG can be written.

(b) Grammars parsed using LR parsers are super set of the class of grammar.

(c) LR parser can detect syntactic error.

(d) LR parser can parse ambiguous grammar.

101. Consider the following grammar G:

$S \rightarrow AB \mid d$

$A \rightarrow aA \mid b$

$B \rightarrow bB \mid c$

The grammar G is

(a) LL(1)

(b) LL(2) but not LL(1)

(c) LR(0)

(d) LR(1) but not LR(0)

102. Consider the following CFG:

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

$T \rightarrow (T * F) \mid F$

$F \rightarrow id$

Which of the following strings are generated by above CFG?

(a) id + id + id (b) id * id * id

(c) (id * id + id) (d) id + id * id

103. Which of the following could result in shift reduce conflict in LR(0) parser?

(a) {A → bc., C → c.}

(b) {A → a.b. B → b.}

(c) {B → .a, A → ∈}

(d) {A → b.c B → b.}

104. Consider the following grammar:

$E \rightarrow E(T) \mid T$

$T \rightarrow T * F \mid id$

$F \rightarrow (id)$

Which of the following are incorrect handle in bottom up parsing for the above grammar?

(a) id * id (b) (id)

(c) id * F (d) id

105. Consider the following SDT:

$E \rightarrow E + T \{E.val = E_1.val + T.val\}$

$E \rightarrow E * T \{E.val = E_1.val * T.val\}$

$E \rightarrow E - E \{E.val = E_1.val - E_2.val\}$

$E \rightarrow id \{E.val = id.num\}$

The above SDT is

(a) S-atrributed

(b) L-attributed

(c) SDT given is incorrect

(d) None of these

106. Which of the following grammars are ambiguous?

(a) $F \to I \mid (E)$
$I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
$T \to F \mid T * F$
$E \to T \mid E + T$

(b) $S \to AB \mid CD$
$A \to 0A1 \mid 01$
$B \to 2B \mid 2$
$C \to OC \mid 0$
$D \to 1D2 \mid 12$

(c) $S \to SS \mid ab$
$A \to B \mid C$
$C \to deG \mid \in$
$G \to C$

(d) None of these

107. Consider the following grammar:

$S \to XY$
$X \to (s) \mid eM$
$Y \to pS \mid \in$
$M \to qX \mid \in$

For the above CFG, LL(1) passing table constructed is as follows:

	(	)	e	p	q	$
S			$S \to XY$			
X	$X \to (s)$		$X \to eM$			
Y				$S \to pS$		
M				$M \to \in$	$M \to qX$	

Which of the following production rules whose entries are left in the parsing table?

(a) $S \to XY$ (b) $Y \to \in$
(c) $M \to \in$ (d) $Y \to \in pS \mid \in$

Try Yourself

T1. Let G be the CFG, l be the number of left most derivations, r be the number of right most derivations and P be the number of parse trees. Assume l, r and P are computed for a particular string. For a given CFG 'G', and given string 'w', what is the relation between l, P and r?

(a) $l \le P \ge r$ (b) $l = P = r$
(c) $l \ge P \le r$ (d) None of these

[Ans: (b)]

T2. Consider the following SDT:

$E_1 \to E_2 + T$	$\{E_1.val = E_2.val + T.val\}$
$E_1 \to T$	$\{E_1.val = T.val\}$
$T_1 \to T_2 * F$	$\{T_1.val = T_2.val * F.val\}$
$F \to (E)\ G$	$\{F.val = 1;\ E.val = G.val\}$
$F \to num$	$\{F.val = num.val\}$
$G \to num$	$\{G.val = num.val\}$
$T \to F$	$\{T.val = F.val\}$

The above SDT is

(a) L-attributed (b) S-attributed
(c) Both (a) and (b) (d) Neither (a) nor (b)

[Ans: (d)]

T3. Which of the following statement is invalid?

(a) Any finite language can be defined by a regular expression.
(b) Attribute grammars can specify the language that can not be specified using a context free grammar.
(c) A recursive descent parser can not directly use a grammar that has right recursive rules.
(d) Both (b) and (c) **[Ans: (d)]**

T4. Consider the following CFG:

stmt $\to \varepsilon \mid$ stmt; stmt; $\mid$ ifstmt $\mid$ whilestmt
ifstmt $\to$ If bexpr THEN stmt $\mid$ IF bexpr THEN stmt ELSE stmt END
whilestmt $\to$ WHILE bexpr DO stmt END
bexpr $\to$ TRUE $\mid$ FALSE

Where ε, ; , IF, THEN, END, WHILE, DO, END, TRUE, and FALSE are terminals, and stmt, ifstmt, whilestmt and bexpr are non-terminals.

The above grammar is

(a) Ambiguous
(b) Unambiguous but not LL(1)
(c) LL(1)
(d) LR(1)

[Ans: (a)]

T5. Consider the following sets of items for some CFG produced while constructing LR (1) parser.

Set 1 (State): A → a., {b}
B → ba., {b,c}

Set 2 (State): A → B.a, {$}
B → aB., {b}

Each item has look-a-head part which is computed using CLR (1) parser. Assume the grammar has follow sets as:

Follow (A) = {a, b} and Follow (B) = {b, c}

Which of the following can be concluded from above two states of CLR (1) construction. [All other states are not in conflict]

(a) Grammar is SLR (1)
(b) Grammar is CLR (1) but not SLR (1)
(c) Grammar is not CLR (1) due to SR conflict
(d) Grammar is not CLR (1) due to RR conflict

[Ans: (d)]

T6. Consider the following augmented grammar with labels a and b

S′ → S
a: S → (S)S
b: S → ε

LR (0) sets are given as following for the above grammar.

(1)	(2)	(3)	(4)	(5)	(6)
S′ → .S S → .(S) S S → .	S′ → S.	S → (.S) S S → .(S) S S → .	S → (S.) S	S → (S). S S → .(S) S S → .	S → (S) S.

If the following SLR (1) table is constructed as below then find the missing entries at E_1, E_2 and E_3 respectively.

	(	)	$	S
1	S_3	r_b	r_b	2
2			Accept	
3	S_3	E_1	E_2	4
4		S_5		
5	S_3	r_b	r_b	E_3
6		r_a	r_a	

(a) r_a, r_b, 5 (b) r_a, r_a, 6
(c) r_b, r_a, 5 (d) r_b, r_b, 6

[Ans: (d)]

T7. For a context-free grammar, FOLLOW(A) is the set of terminals that can appear immediately to the right of non-terminal A in some "sentential" form. We defined to sets LFOLLOW(A) and RFOLLOW(A) by replacing the word "left most sentential" and "right most sentential" respectively in the definition of FOLLOW(A). Which of the following statements is/are true?

(a) FOLLOW(A) and RFOLLOW(A) may be different.
(b) FOLLOW(A) and RFOLLOW(A) are always the same.
(c) All the three sets are identical.
(d) All the three sets are different.

[Ans: (c)]

T8. Which of the following statements is false?

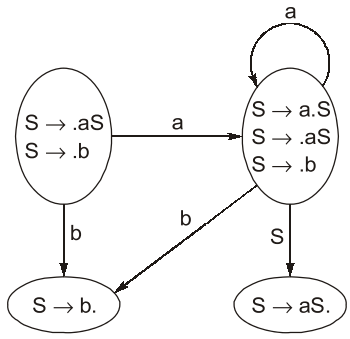
(a) an unambiguous grammar has same leftmost and rightmost derivation
(b) an LL(1) parser is a top-down parser
(c) LALR is more powerful than SLR
(d) an ambiguous grammar can never be LR(k) for any k

[Ans: (a)]

T9. Consider the production.

S → aS | b

The parsing automation is below:



Which of the following stack contents causes the parser to redeuce by some production?

(a) aaS (b) a
(c) aa (d) bb

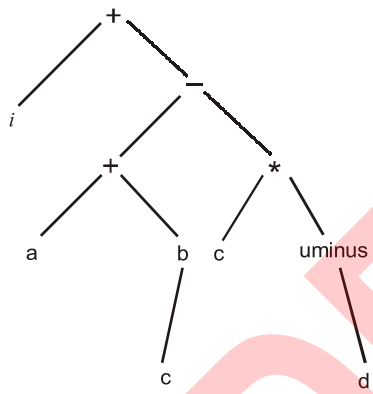
[Ans: (a)]

■■■■

3 Intermediate Code Generation and Code Optimization

Multiple Choice Questions

Q.1 The intermediate code generated for the following syntax tree is



(a) iac + cdb + − +
(b) iacb + cd uminus ∗ − +
(c) iacb + cd ∗ − +
(d) None of the above

Q.2 Which is not a three address code?
(a) If $x < y$ go to L
(b) $x = y$ [I]
(c) $x = \& y$
(d) None of the above

Q.3 Generation of intermediate code based on an abstract machine model is useful in compilers because
(a) it makes implementation of lexical analysis and syntax analysis easier
(b) syntax-directed translations can be written for intermediate code generation
(c) it enhances the portability of the front end of the compiler
(d) it is not possible to generate code for real machines directly from high level language programs

[GATE-1994]

Q.4 'Three address code' technique for intermediate code generation shows that each statement usually contains three addresses. Three addresses are as follows
(a) two addresses for operands, one for operator
(b) one for all operator, one for all operands and one for result
(c) one for result, two for operands
(d) None of the above

Q.5 The following code has

```
for (i = 0; i < n; i++)
{
    S(i);
}
```

(a) Common sub expression
(b) Dead code
(c) Loop invariant
(d) Loop unrolling

Q.6 Consider the following arithmetic infix expression Q.

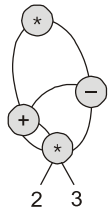
A + (B ∗ C –(D/E ↑ F) ∗ G) ∗ H.

Transform 'Q' into its equivalent postfix expression.
(a) ABC ∗ DE ↑ F/G ∗ –H∗ +
(b) ABC ∗ DEF ↑ G ∗ H – ∗ +

(c) ABC * DEF ↑ /G * –H * +
(d) None of the above

Q.7 Which of the following is not a type of three address statements?
(a) copy statements
(b) index assignment statements
(c) pointer assignments
(d) none of the above

Q.8 Consider the following DAG (Directed Acyclic Graph)



Identify the equivalent 3-address code for the above DAG?

(a) $t_1 = 2 * 3$
$t_2 = t_1 * t_1$
$t_3 = t_2 + t_2$
$t_4 = t_3 - t_2$

(b) $t_1 = 2 * 3$
$t_2 = t_1 + t_1$
$t_3 = t_2 - t_1$
$t_4 = t_2 * t_3$

(c) $t_1 = 2 * 3$
$t_2 = t_1 * t_1$
$t_3 = t_2 - t_1$
$t_4 = t_2 + t_3$

(d) Given graph is not DAG

Q.9 Consider the following three address code table.

Operation	Operand-1	Operand-2	Result
–	c		t_1
×	b	t_1	t_2
+	a	t_2	t_3
Move	t_3		a

Which of the following expression represents the above three address code (quadruple notation)?

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

Q.10 Consider the following intermediate program in three address code

$p = a - b$
$q = p * c$
$p = u * v$
$q = p + q$

Which one of the following corresponds to a static single assignment form of the above code?

(a) $p_1 = a - b$
$q_1 = p_1 * c$
$p_1 = u * v$
$q_1 = p_1 + q_1$

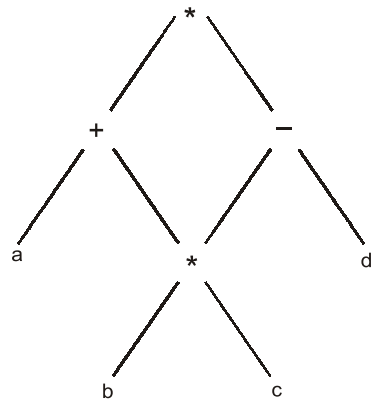
(b) $p_3 = a - b$
$q_4 = p_3 * c$
$p_4 = u * v$
$q_5 = p_4 + q_4$

(c) $p_1 = a - b$
$q_1 = p_2 * c$
$p_3 = u * v$
$q_2 = p_4 + q_3$

(d) $p_1 = a - b$
$q_1 = p * c$
$q_2 = u * v$
$q_2 = p + q$

[GATE-2017]

Q.11 The DAG shown here represents:



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

Q.12 Any Directed Acyclic Graph must have
(a) atleast one vertex with indegree 1 and atleast one vertex with outdegree 0
(b) atleast one vertex with indegree 0 or atleast one vertex with outdegree 0

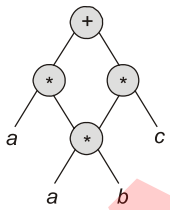
(c) atleast one vertex with indegree 0 and atleast one vertex with outdegree 0
(d) atleast one vertex with indegree 0 and atleast one vertex with outdegree 1

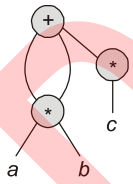
Q.13 A directed acyclic graph represents one form of intermediate representation. The number of non terminals nodes in DAG of $a=(b+c)*(b+c)$ expression is
(a) 2 (b) 3
(c) 4 (d) 5

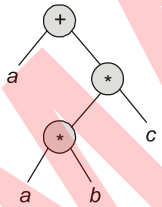
Q.14 A pictorial representation of the value computed by each statement in the basic block is
(a) tree (b) DAG
(c) graph (d) None of these

Q.15 DAG representation of a basic block allows
(a) automatic detection of local common sub-expressions
(b) automatic detection of induction variables
(c) automatic detection of loop invariant
(d) None of the above

Q.16 The DAG for expression $a + b + a * b * c$ is

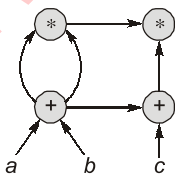
(a) 

(b) 

(c) 

(d) None of these

Q.17 Consider the following Directed Acyclic Graph :



Which one of the following expressions correctly represents the above graph?
(a) $(a + b) * (a + b) * (a + b) + c$
(b) $(a + b + c) * (a + b + c)$
(c) $[(a + b) * (a + b)] * [(a + b) + c]$
(d) None of these

Q.18 Consider the intermediate code given below:

1. $i = 1$;
2. if $i \le n$ goto 4
3. goto 11
4. $t_1 = a + b$;
5. $S = t_1$
6. $t_2 = t_1 * c$
7. $S = t_2$
8. $t_3 = t_2 + d$
9. $S = t_3$
10. $i = i + 1$ goto (2)
11. end

Which of the following code has above intermediate code?

(a)
```
i = 1
while (i ≤ n)
{
   S = a+b*c+d
   i++
}
```

(b)
```
i = 1
while (i ≤ n)
{
   S = a + b
   S = b * c
   S = c + d
   i++
}
```

(c)
```
i = 1
while (i ≤ n)
{
   S = a + b
   S = S * c
   S = S + d
   i++
}
```

(d) None of these

Q.19 For a C program accessing X[i][j][k], the following intermediate code is generated by a compiler. Assume that the size of an integer is 32 bits and the size of a character is 8 bits.

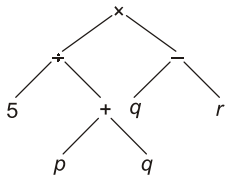
$t_0 = i * 1024$
$t_1 = j * 32$
$t_2 = k * 4$
$t_3 = t_1 + t_0$
$t_4 = t_3 + t_2$
$t_5 = X[t_4]$

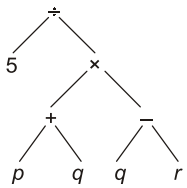
Which one of the following statements about the source code for the C program is CORRECT?

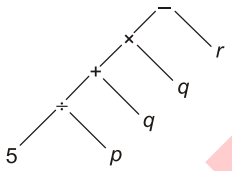
(a) X is declared as "int X[32][32][8]".
(b) X is declared as "int X[4][1024][32]".
(c) X is declared as "char X[4][32][8]".
(d) X is declared as "char X[32][16][2]".

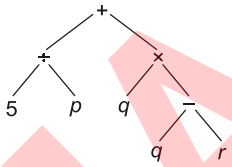
[GATE-2014 (Set-2)]

Q.20 Consider the expression $[5 \div (p + q) \times (q - r)]$. Assume that $\div$ has lowest precedence than $+$, $-$ and $\times$, $+$ and $-$ has same precedence but more than $\times$. Which of the following is the corresponding parse tree?

(a)


(b)


(c)


(d)


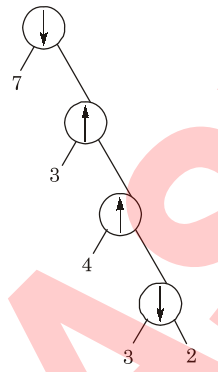
Q.21 Consider the following code segment:

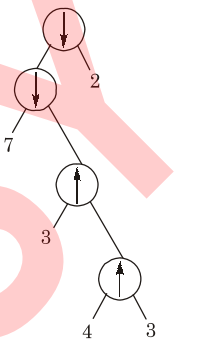
$a = b + c$
$k = a - d$
$a = k + b$
$u = v + w$
$u = a - u$

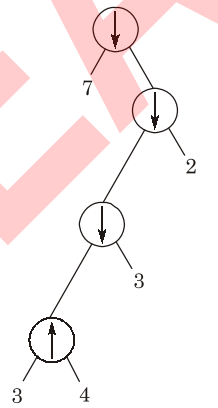
The minimum number of total variables required to convert the above code segment to static single assignment from is

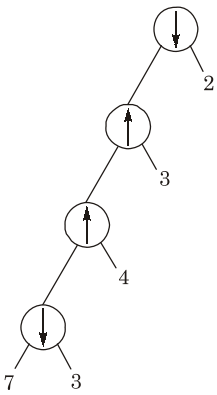
(a) 6 (b) 10
(c) 8 (d) 9

Q.22 Consider two binary operators '↑' and '↓' with the precedence of operator ↓ being lower than that of the operator ↑. Operator ↑ is right associative while operator ↓ is left associative. Which one of the following represents the parse tree for expression (7 ↓ 3 ↑ 4 ↑ 3 ↓ 2)

(a)


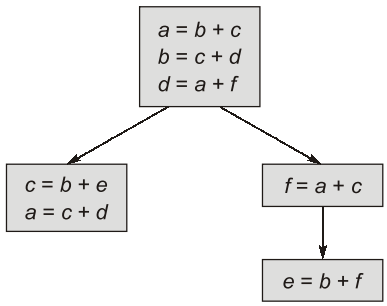
(b)


(c)


(d)


[GATE-2011]

Q.23 Consider the following control flow graph, the variables that are live-in and live-out at block-3 are?



(a) {a, c} and {f}
(b) {a, b, c} and {b, d, f}
(c) {b, d, e, f} and {b, c, d, e, f}
(d) {a, b, c} and {b, f}

Numerical Answer Type Questions

Q.24 Consider the intermediate code given below:

1. $i = 1$
2. $j = 1$
3. $t_1 = 5 * i$
4. $t_2 = t_1 + j$
5. $t_3 = 4 * t_2$
6. $t_4 = t_3$
7. $a[t_4] = -1$
8. $j = j + 1$
9. if $j <= 5$ goto (3)
10. $i = i + 1$
11. if $i < 5$ goto (2)

The number of nodes and edges in the control-flow-graph constructed for the above code, respectively, are _____.

Q.25 Consider the following 3-address code.

$t_1 = t + e$
$t_2 = g + a$
$t_3 = t_1 * t_2$
$t_4 = t_2 + t_2$
$t_5 = t_4 + t_3$

There are five temporary variables in above code. Find minimum number of temporary variables can be used in the equivalent optimized 3-address code of above code.

Q.26 The least number of temporary variables required to create a three-address code in static single assignment form for the expression $q + r/3 + s - t * 5 + u * v/w$ is _____.

Q.27 Find the minimum number of temporary variables are created in 3-address code for the following expression

$a + b * c + d - e - a + b * c$

Consider following grammar for precedence and associativity.

$S \rightarrow ES \mid \in$
$E \rightarrow E - F \mid F$
$F \rightarrow F + G \mid H$
$G \rightarrow G * H \mid H$
$H \rightarrow id \mid \in$

Q.28 Find the minimum number of nodes and edges in the DAG for the following code segment

$a = b + c$
$b = a[i]$
$b[i] = x$

Q.29 Consider the following expression:

$x + x + x + x + x + x + x + x + x$

How many minimum edges are present in the DAG that represents above expression. [DAG is Directed Acyclic Graph].

Multiple Select Questions

Q.30 Which of the following are the applications of DAG (Directed Acyclic Graph)?

(a) Determining the common sub-expressions.
(b) Determining which names are used inside the block and computed outside the block.
(c) Determining which statements of the block could have their computed value outside the block.
(d) Simplifying the list of quadruples by eliminating the common sub-expressions and not performing the assignment of the form $x = y$ unless and until it is a must.

■■■■

4 Runtime Environment

Multiple Choice Questions

Q.1 Which of the following is/are TRUE?

1. For any compiler generated temporaries the space is allocated in run-time heap only.
2. Programming languages which allow recursion require a stack-based allocation scheme and cannot be implemented with static storage allocation scheme.
3. For any dynamically allocated global variable the compiler might allocate space either in static area or in the activation record.
4. For a variable local to a procedure the space can be allocated in activation record.

(a) 1 and 3 (b) 2 and 4
(c) 1 and 4 (d) 4 only

[DRDO-2009]

Q.2 Which of the following statements is FALSE?

(a) In statically typed language, each variable in a program has a fixed type
(b) In un-typed languages, values do not have any types
(c) In dynamically typed languages, variables have no types
(d) In all statically typed languages, each variable in a program is associated with values of only a single type during the execution of the program

[GATE 2003]

Q.3 Which of the following statements are TRUE?

I. There exist parsing algorithms for some programming languages whose complexities are less than $O(n^3)$.
II. A programming language which allows recursion can be implemented with static storage allocation.
III. No L-attributed definition can be evaluated in the framework of bottom-up parsing.
IV. Code improving transformations can be performed at both source language and intermediate code level.

(a) I and II (b) I and IV
(c) III and IV (d) I, III and IV

[GATE 2009]

Q.4 Which of the following are true?

1. A programming language which does not permit global variables of any kind and has no nesting of procedures/functions, but permits recursion can be implemented with static storage allocation.
2. Multi-level access link (or display) arrangement is needed to arrange activation records only if the programming language being implemented has nesting of procedures/function.
3. Recursion in programming languages cannot be implemented with dynamic storage allocation.
4. Nesting of procedures/functions and recursion require a dynamic heap allocation scheme and cannot be implemented with a stack-based allocation scheme for activation records.
5. Programming languages which permit a function to return a function as its result cannot be implemented with a stack-based storage allocation scheme for activation records.

(a) 2 and 5 only
(b) 1, 3 and 4 only
(c) 1, 2 and 5 only
(d) 2, 3 and 5 only

[GATE-2008]

Q.5 Choose the false statements.
1. Control stack keeps track of live procedure activations.
2. Activation records can be managed with the help of stack.
3. Dangling reference is a reference to a storage that has been deallocated.

(a) 1 and 2
(b) 2 and 3
(c) 1 and 3
(d) None of these

Q.6 Which of the following storage(s) can grow and shrink at runtime?
(a) Stack section
(b) Heap section
(c) Both (a) and (b)
(d) None of these

Q.7 Which of the following may need to save or restore when a function call or return happens at runtime.
(a) Control link
(b) Program Counter, Register values
(c) Access link
(d) All of the above

Q.8 What is the purpose of access link field which is used in activation record?
(a) to access the non-local data
(b) to return the value to the caller
(c) to access the program counter information
(d) None of these

Q.9 Which of the following parameter is not included in the activation record of recursive function call?
(a) Local variables of function
(b) Return value of function
(c) Global variables of program
(d) Access link

Q.10 A garbage is
(a) unallocated storage
(b) allocated storage with all access paths to it destroyed
(c) allocated storage
(d) uninitialized storage

Multiple Select Questions

Q.11 Which of the following statements is/are correct?
(a) References are resolved during link time.
(b) Relocation of the program done at loading time.
(c) Activation record created during run time.
(d) Activation record created during compile time.

Q.12 Activation record content are
(a) Static link
(b) Saved registers
(c) Return value
(d) Scratch area

Try Yourself

T1. Consider the following fields:
F1 : Frame pointer link
F2 : Access link
F3 : Local variable
F4 : Return address
Which of the above fields present in the activation record?
(a) F1, F2
(b) F2, F3, F4
(c) F1, F2, F4
(d) F1, F2, F3, F4

[Ans: (b)]

T2. Consider the program given below, in a block-structured pseudo-language with lexical scoping and nesting of procedures permitted.

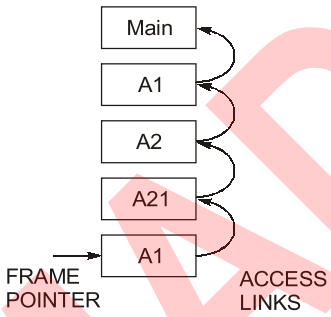
```
Program main;
Var ...
Procedure A1;
        Var ...
        Call A2;
End A1
Procedure A2;
        Var ...
        Procedure A21;
                Var ...
                Call A1,
        End A21
        Call A21;
End A2
Call A1;
End main;
```

Consider the calling chain:

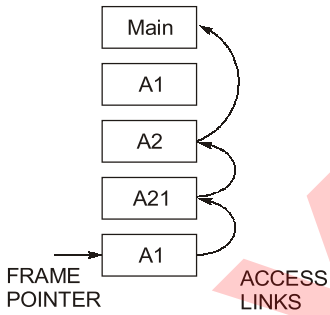
Main → A1 → A2 → A21 → A1

The correct set of activation records along with their access links is given by

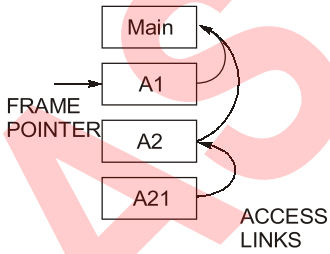
(a)



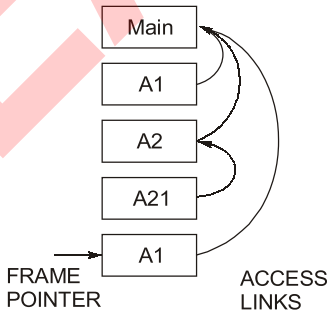
(b)



(c)



(d)



[Ans: (d)]

■■■■