2 Compiler Design (186)

Lexical analysis, Parsing, Syntax-directed translation, Runtime environments, Intermediate code generation.

Mark Distribution in Previous GATE

Year	2019	2018	2017-1	2017-2	2016-1	2016-2	Minimum	Average	Maximum
1 Mark Count	2	1	2	2	1	1	1	1.5	2
2 Marks Count	2	2	2	1	3	2	1	2	3
Total Marks	6	5	6	4	7	5	**4**	**5.5**	**6**

2.1 Abstract Syntax Tree (1)

2.1.1 Abstract Syntax Tree: GATE2015-2-14

https://gateoverflow.in/8084

In the context of abstract-syntax-tree (AST) and control-flow-graph (CFG), which one of the following is TRUE?

A. In both AST and CFG, let node N_2 be the successor of node N_1. In the input program, the code corresponding to N_2 is present after the code corresponding to N_1
B. For any input program, neither AST nor CFG will contain a cycle
C. The maximum number of successors of a node in an AST and a CFG depends on the input program
D. Each node in AST and CFG corresponds to at most one statement in the input program

gate2015-2 compiler-design easy abstract-syntax-tree

2.2 Assembler (7)

2.2.1 Assembler: GATE1992-01,viii

https://gateoverflow.in/553

The purpose of instruction location counter in an assembler is _____

gate1992 compiler-design assembler normal

2.2.2 Assembler: GATE1992-03,ii

https://gateoverflow.in/579

Mention the pass number for each of the following activities that occur in a two pass assembler:

A. object code generation
B. literals added to literal table
C. listing printed
D. address resolution of local symbols

gate1992 compiler-design assembler easy

2.2.3 Assembler: GATE1992-3,i

https://gateoverflow.in/578

Write short answers to the following:

(i). Which of the following macros can put a macro assembler into an infinite loop?

```
.MACRO MI,X
.IF EQ,X
M1 X+1
.ENDC
.IF NE,X
.WORD X
.ENDC
.ENDM
```

```
.MACRO M2,X
.IF EQ,X
M2 X
.ENDC
.IF NE,X
.WORD X+1
.ENDC
.ENDM
```

Give an example call that does so.

gate1992 compiler-design assembler normal

2.2.4 Assembler: GATE1993-7.6

https://gateoverflow.in/2294

A simple two-pass assembler does the following in the first pass:

A. It allocates space for the literals.
B. It computes the total length of the program.
C. It builds the symbol table for the symbols and their values.

D. It generates code for all the load and store register instructions.
E. None of the above.

gate1993 compiler-design assembler easy

2.2.5 Assembler: GATE1994-17

https://gateoverflow.in/2513

State whether the following statements are True or False with reasons for your answer:

A. Coroutine is just another name for a subroutine.
B. A two pass assembler uses its machine opcode table in the first pass of assembly.

gate1994 compiler-design normal assembler

2.2.6 Assembler: GATE1994-18

https://gateoverflow.in/2514

State whether the following statements are True or False with reasons for your answer

A. A subroutine cannot always be used to replace a macro in an assembly language program.
B. A symbol declared as 'external' in an assembly language program is assigned an address outside the program by the assembler itself.

gate1994 compiler-design normal assembler true-false

2.2.7 Assembler: GATE1996-1.17

https://gateoverflow.in/2721

The pass numbers for each of the following activities

i. object code generation
ii. literals added to literal table
iii. listing printed
iv. address resolution of local symbols that occur in a two pass assembler

respectively are

A. $1,2,1,2$ B. $2,1,2,1$ C. $2,1,1,2$ D. $1,2,2,2$

gate1996 compiler-design normal assembler

2.3 Code Optimization (4)

2.3.1 Code Optimization: GATE2008-12

https://gateoverflow.in/410

Some code optimizations are carried out on the intermediate code because

A. They enhance the portability of the compiler to the target processor
B. Program analysis is more accurate on intermediate code than on machine code
C. The information from dataflow analysis cannot otherwise be used for optimization
D. The information from the front end cannot otherwise be used for optimization

gate2008 normal code-optimization compiler-design

2.3.2 Code Optimization: GATE2014-1-17

https://gateoverflow.in/1784

Which one of the following is **FALSE**?

A. A basic block is a sequence of instructions where control enters the sequence at the beginning and exits at the end.
B. Available expression analysis can be used for common subexpression elimination.
C. Live variable analysis can be used for dead code elimination.
D. $x = 4 * 5 \Rightarrow x = 20$ is an example of common subexpression elimination.

gate2014-1 compiler-design code-optimization normal

2.3.3 Code Optimization: GATE2014-3-11

https://gateoverflow.in/2045

The minimum number of arithmetic operations required to evaluate the polynomial $P(X) = X^5 + 4X^3 + 6X + 5$ for a given value of X, using only one temporary variable is _____.

gate2014-3 compiler-design numerical-answers normal code-optimization

2.3.4 Code Optimization: GATE2014-3-34

https://gateoverflow.in/2068

Consider the basic block given below.

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

The minimum number of nodes and edges present in the DAG representation of the above basic block respectively are

A. 6 and 6 B. 8 and 10 C. 9 and 12 D. 4 and 4

gate2014-3 compiler-design code-optimization normal

2.4 Compilation Phases (8)

2.4.1 Compilation Phases: GATE1987-1-xi

https://gateoverflow.in/80282

In a compiler the module the checks every character of the source text is called:

A. The code generator.
B. The code optimiser.
C. The lexical analyser.
D. The syntax analyser.

gate1987 compiler-design compilation-phases

2.4.2 Compilation Phases: GATE1990-2-ix

https://gateoverflow.in/84033

Match the pairs in the following questions:

(a)	Lexical analysis	(p)	DAG's
(b)	Code optimization	(q)	Syntax trees
(c)	Code generation	(r)	Push down automaton
(d)	Abelian groups	(s)	Finite automaton

gate1990 match-the-following compiler-design compilation-phases

2.4.3 Compilation Phases: GATE2005-61

https://gateoverflow.in/4066

Consider line number 3 of the following C-program.

```
int main() {                /*Line 1 */
    int I, N;               /*Line 2 */
    fro (I=0, I<N, I++); /*Line 3 */
}
```

Identify the compiler's response about this line while creating the object-module:

A. No compilation error
B. Only a lexical error
C. Only syntactic errors
D. Both lexical and syntactic errors

gate2005 compiler-design compilation-phases normal

2.4.4 Compilation Phases: GATE2009-17

https://gateoverflow.in/1309

Match all items in Group 1 with the correct options from those given in Group 2.Syntax analysis

Group 1		Group 2	
P.	Regular Expression	1.	Syntax analysis
Q.	Pushdown automata	2.	Code generation
R.	Dataflow analysis	3.	Lexical analysis
S.	Register allocation	4.	Code optimization

A. P-4, Q-1, R-2, S-3
B. P-3, Q-1, R-4, S-2
C. P-3, Q-4, R-1, S-2
D. P-2, Q-1, R-4, S-3

gate2009 compiler-design easy compilation-phases

2.4.5 Compilation Phases: GATE2015-2-19

https://gateoverflow.in/8098

Match the following:

P.	Lexical analysis	1.	Graph coloring
Q.	Parsing	2.	DFA minimization
R.	Register allocation	3.	Post-order traversal
S.	Expression evaluation	4.	Production tree

A. P-2, Q-3, R-1, S-4
B. P-2, Q-1, R-4, S-3
C. P-2, Q-4, R-1, S-3
D. P-2, Q-3, R-4, S-1

gate2015-2 compiler-design normal compilation-phases

2.4.6 Compilation Phases: GATE2016-2-19

https://gateoverflow.in/39548

Match the following:

(P)	Lexical analysis	(i)	Leftmost derivation
(Q)	Top down parsing	(ii)	Type checking
(R)	Semantic analysis	(iii)	Regular expressions
(S)	Runtime environment	(iv)	Activation records

A. P $\leftrightarrow$ i, Q $\leftrightarrow$ ii, R $\leftrightarrow$ iv, S $\leftrightarrow$ iii
B. P $\leftrightarrow$ iii, Q $\leftrightarrow$ i, R $\leftrightarrow$ ii, S $\leftrightarrow$ iv
C. P $\leftrightarrow$ ii, Q $\leftrightarrow$ iii, R $\leftrightarrow$ i, S $\leftrightarrow$ iv
D. P $\leftrightarrow$ iv, Q $\leftrightarrow$ i, R $\leftrightarrow$ ii, S $\leftrightarrow$ iii

gate2016-2 compiler-design easy compilation-phases

2.4.7 Compilation Phases: GATE2017-2-05

https://gateoverflow.in/118592

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

P. Syntax tree	i. Code generator
Q. Character stream	ii. Syntax analyser
R. Intermediate representation	iii. Semantic analyser
S. Token stream	iv. Lexical analyser

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

gate2017-2 compiler-design compilation-phases easy

2.4.8 Compilation Phases: GATE2018-8

https://gateoverflow.in/204082

Which one of the following statements is FALSE?

A. Context-free grammar can be used to specify both lexical and syntax rules
B. Type checking is done before parsing
C. High-level language programs can be translated to different Intermediate Representations

D. Arguments to a function can be passed using the program stack

gate2018 compiler-design easy compilation-phases

2.5 Expression Evaluation (2)

2.5.1 Expression Evaluation: GATE2002-2.19

https://gateoverflow.in/849

To evaluate an expression without any embedded function calls

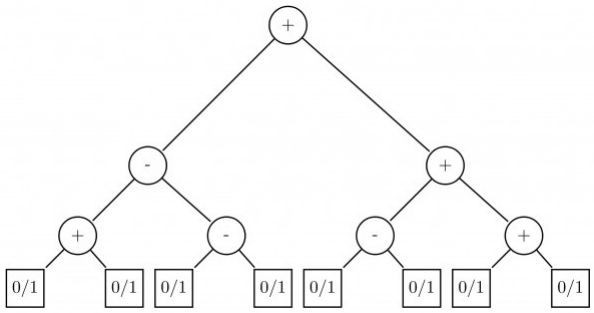
A. One stack is enough
B. Two stacks are needed
C. As many stacks as the height of the expression tree are needed
D. A Turing machine is needed in the general case

gate2002 compiler-design expression-evaluation easy

2.5.2 Expression Evaluation: GATE2014-2-39

https://gateoverflow.in/1999

Consider the expression tree shown. Each leaf represents a numerical value, which can either be 0 or 1. Over all possible choices of the values at the leaves, the maximum possible value of the expression represented by the tree is ___.



gate2014-2 compiler-design normal expression-evaluation numerical-answers

2.6 Grammar (41)

2.6.1 Grammar: GATE1990-16a

https://gateoverflow.in/86869

Show that grammar $G1$ is ambiguous using parse trees:

G_1 : $S \rightarrow$ if S then S else S

$S \rightarrow$ if S then S

gate1990 descriptive compiler-design grammar

2.6.2 Grammar: GATE1991-10a

https://gateoverflow.in/537

Consider the following grammar for arithmetic expressions using binary operators $-$ and $/$ which are not associative

$$E \rightarrow E - T \mid T$$

$$T \rightarrow T/F \mid F$$

$$F \rightarrow (E) \mid id$$

(E is the start symbol)

Is the grammar unambiguous? Is so, what is the relative precedence between $-$ and $/$? If not, give an unambiguous grammar that gives $/$ precedence over $-$.

gate1991 grammar compiler-design normal descriptive

2.6.3 Grammar: GATE1991-10b

https://gateoverflow.in/43604

Consider the following grammar for arithmetic expressions using binary operators $-$ and $/$ which are not associative

$E \rightarrow E - T \mid T$

$T \rightarrow T/F \mid F$

$F \rightarrow (E) \mid id$

(E is the start symbol)

Does the grammar allow expressions with redundant parentheses as in (id/id) or in $id - (id/id)$? If so, convert the grammar into one which does not generate expressions with redundant parentheses. Do this with minimum number of changes to the given production rules and adding at most one more production rule.

gate1991 grammar compiler-design normal descriptive

2.6.4 Grammar: GATE1991-10c

https://gateoverflow.in/43605

Consider the following grammar for arithmetic expressions using binary operators $-$ and $/$ which are not associative

- $E \rightarrow E - T \mid T$
- $T \rightarrow T/F \mid F$
- $F \rightarrow (E) \mid id$

(E is the start symbol)

Does the grammar allow expressions with redundant parentheses as in (id/id) or in $id - (id/id)$? If so, convert the grammar into one which does not generate expressions with redundant parentheses. Do this with minimum number of changes to the given production rules and adding at most one more production rule.

Convert the grammar obtained above into one that is not left recursive.

gate1991 grammar compiler-design normal descriptive

2.6.5 Grammar: GATE1994-1.18

https://gateoverflow.in/2461

Which of the following features cannot be captured by context-free grammars?

A. Syntax of if-then-else statements
B. Syntax of recursive procedures
C. Whether a variable has been declared before its use
D. Variable names of arbitrary length

gate1994 compiler-design grammar normal

2.6.6 Grammar: GATE1994-20

https://gateoverflow.in/2516

A grammar G is in Chomsky-Normal Form (CNF) if all its productions are of the form $A \rightarrow BC$ or $A \rightarrow a$, where A, B and C, are non-terminals and a is a terminal. Suppose G is a CFG in CNF and w is a string in $L(G)$ of length n, then how long is a derivation of w in G?

gate1994 compiler-design grammar normal

2.6.7 Grammar: GATE1994-3.5

https://gateoverflow.in/2482

Match the following items

(i)	Backus-Naur form	(a)	Regular expressions
(ii)	Lexical analysis	(b)	LALR(1) grammar
(iii)	YACC	(c)	LL(1) grammars
(iv)	Recursive descent parsing	(d)	General context-free grammars

gate1994 compiler-design grammar normal

2.6.8 Grammar: GATE1995-1.10

https://gateoverflow.in/2597

Consider a grammar with the following productions

- $S \rightarrow a\alpha b \mid b\alpha c \mid aB$
- $S \rightarrow \alpha S \mid b$
- $S \rightarrow \alpha bb \mid ab$
- $S\alpha \rightarrow bdb \mid b$

The above grammar is:

A. Context free B. Regular C. Context sensitive D. $LR(k)$

gate1995 compiler-design grammar normal

2.6.9 Grammar: GATE1995-9

https://gateoverflow.in/2644

A. Translate the arithmetic expression $a^* - (b + c)$ into syntax tree.
B. A grammar is said to have cycles if it is the case that

$A \Rightarrow^+ A$

Show that no grammar that has cycles can be LL(1).

gate1995 compiler-design grammar normal

2.6.10 Grammar: GATE1996-11

https://gateoverflow.in/2763

Let G be a context-free grammar where $G = (\{S, A, B, C\}, \{a, b, d\}, P, S)$ with the productions in P given below.

$S \rightarrow ABAC$

$A \rightarrow aA \mid \varepsilon$

$B \rightarrow bB \mid \varepsilon$

$C \rightarrow d$

(ε denotes the null string). Transform the grammar G to an equivalent context-free grammar G' that has no ε productions and no unit productions. (A unit production is of the form $x \rightarrow y$, and x and y are non terminals).

gate1996 compiler-design grammar normal

2.6.11 Grammar: GATE1996-2.10

https://gateoverflow.in/2739

The grammar whose productions are

- $\langle \text{stmt} \rangle \rightarrow \text{if id then } \langle \text{stmt} \rangle$
- $\langle \text{stmt} \rangle \rightarrow \text{if id then } \langle \text{stmt} \rangle \text{ else } \langle \text{stmt} \rangle$
- $\langle \text{stmt} \rangle \rightarrow \text{id} := \text{id}$

is ambiguous because

(a) the sentence

```
if a then if b then c:= d
```

has more than two parse trees

(b) the left most and right most derivations of the sentence

```
if a then if b then c:= d
```

give rise to different parse trees

(c) the sentence

```
if a then if b then c:= d else c:= f
```

has more than two parse trees

(d) the sentence

```
if a then if b then c:= d else c:= f
```

has two parse trees

gate1996 compiler-design grammar normal

2.6.12 Grammar: GATE1997-1.6

https://gateoverflow.in/2222

In the following grammar

$$\begin{aligned} X &::= X \oplus Y \mid Y \\ Y &::= Z * Y \mid Z \\ Z &::= id \end{aligned}$$

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

gate1997 compiler-design grammar normal

2.6.13 Grammar: GATE1997-11

https://gateoverflow.in/2271

Consider the grammar

- $S \rightarrow bSe$
- $S \rightarrow PQR$
- $P \rightarrow bPc$
- $P \rightarrow \varepsilon$
- $Q \rightarrow cQd$
- $Q \rightarrow \varepsilon$
- $R \rightarrow dRe$
- $R \rightarrow \varepsilon$

where S,P,Q,R are non-terminal symbols with S being the start symbol; b,c,d,e are terminal symbols and 'ε' is the empty string. This grammar generates strings of the form b^i,c^j,d^k,e^m for some $i,j,k,m \geq 0$.

a. What is the condition on the values of i,j,k,m?
b. Find the smallest string that has two parse trees.

gate1997 compiler-design grammar normal theory-of-computation

2.6.14 Grammar: GATE1998-14

https://gateoverflow.in/1728

A. Let $G_1 = (N,T,P,S_1)$ be a CFG where, $N = \{S_1, A, B\}, T = \{a,b\}$ and P is given by

$S_1 \rightarrow aS_1b$	$S_1 \rightarrow aBb$
$S_1 \rightarrow aAb$	$B \rightarrow Bb$
$A \rightarrow aA$	$B \rightarrow b$
$A \rightarrow a$	

What is $L(G_1)$?

B. Use the grammar in Part(a) to give a CFG for $L_2 = \{a^ib^ja^kb^l \mid i,j,k,l \geq 1, i = j \text{ or } k = l\}$ by adding not more than 5 production rules.

C. Is L_2 inherently ambiguous?

gate1998 compiler-design grammar descriptive

2.6.15 Grammar: GATE1998-6b

https://gateoverflow.in/1697

Consider the grammar

- S $\rightarrow Aa \mid b$
- A $\rightarrow Ac \mid Sd \mid \epsilon$

Construct an equivalent grammar with no left recursion and with minimum number of production rules.

gate1998 compiler-design grammar descriptive

2.6.16 Grammar: GATE1999-2.15

https://gateoverflow.in/1493

A grammar that is both left and right recursive for a non-terminal, is

A. Ambiguous
B. Unambiguous
C. Information is not sufficient to decide whether it is ambiguous or unambiguous
D. None of the above

gate1999 compiler-design grammar normal

2.6.17 Grammar: GATE2000-2.21, ISRO2015-24

https://gateoverflow.in/668

Given the following expression grammar:

$$E \rightarrow E * F \mid F + E \mid F$$

$$F \quad \rightarrow F - F \mid id$$

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 $*$

gate2000 grammar normal compiler-design isro2015

2.6.18 Grammar: GATE2001-1.18

https://gateoverflow.in/711

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

gate2001 compiler-design grammar normal

2.6.19 Grammar: GATE2001-18

https://gateoverflow.in/759

A. Remove left-recursion from the following grammar: $S \rightarrow Sa \mid Sb \mid a \mid b$
B. Consider the following grammar:

$$S \rightarrow aSbS \mid bSaS \mid \epsilon$$

Construct all possible parse trees for the string abab. Is the grammar ambiguous?

gate2001 compiler-design grammar descriptive

2.6.20 Grammar: GATE2003-56

https://gateoverflow.in/944

Consider the grammar shown below

- $S \rightarrow iEtSS' \mid a$
- $S' \rightarrow eS \mid \epsilon$

- $E \rightarrow b$

In the predictive parse table, M, of this grammar, the entries $M[S', e]$ and $M[S', \$]$ respectively are

A. $\{S' \rightarrow eS\}$ and $\{S' \rightarrow \epsilon\}$
B. $\{S' \rightarrow eS\}$ and $\{\}$
C. $\{S' \rightarrow \epsilon\}$ and $\{S' \rightarrow \epsilon\}$
D. $\{S' \rightarrow eS, S' \rightarrow \varepsilon\}$ and $\{S' \rightarrow \epsilon\}$

gate2003 compiler-design grammar normal

2.6.21 Grammar: GATE2003-58

https://gateoverflow.in/946

Consider the translation scheme shown below.

S $\rightarrow$ T R

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

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. $+ + 9\,5\,2$

gate2003 compiler-design grammar normal

2.6.22 Grammar: GATE2004-45

https://gateoverflow.in/1042

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

$$\begin{array}{ll} E \rightarrow E_1 \# T & \{E.value = E_1.value * T.value\} \\ \quad |T & \{E.value = T.value\} \\ T \rightarrow T_1 \& F & \{T.value = T_1.value + F.value\} \\ \quad |F & \{T.value = F.value\} \\ F \rightarrow \text{num} & \{F.value = num.value\} \end{array}$$

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

gate2004 compiler-design grammar normal

2.6.23 Grammar: GATE2004-8

https://gateoverflow.in/1005

Which of the following grammar rules violate the requirements of an operator grammar? P, Q, R are nonterminals, and r, s, t are terminals.

I. P $\rightarrow$ Q R
II. P $\rightarrow$ Q s R
III. P $\rightarrow$ ε
IV. P $\rightarrow$ Q t R r

A. (I) only B. (I) and (III) only C. (II) and (III) only D. (III) and (IV) only

gate2004 compiler-design grammar normal

2.6.24 Grammar: GATE2004-88

https://gateoverflow.in/1082

Consider the following grammar G:

$S \rightarrow bS \mid aA \mid b$

$A \rightarrow bA \mid aB$

$B \rightarrow bB \mid aS \mid a$

Let $N_a(w)$ and $N_b(w)$ denote the number of a's and b's in a string ω respectively.

The language $L(G)$ over $\{a,b\}^+$ generated by G is

A. $\{w \mid N_a(w) > 3N_b(w)\}$
B. $\{w \mid N_b(w) > 3N_a(w)\}$
C. $\{w \mid N_a(w) = 3k, k \in \{0,1,2,\ldots\}\}$
D. $\{w \mid N_b(w) = 3k, k \in \{0,1,2,\ldots\}\}$

gate2004 compiler-design grammar normal

2.6.25 Grammar: GATE2005-59

https://gateoverflow.in/1382

Consider the grammar:

$$E \rightarrow E + n \mid E \times n \mid n$$

For a sentence $n + n \times n$, the handles in the right-sentential form of the reduction are:

A. $n, E + n$ and $E + n \times n$
B. $n, E + n$ and $E + E \times n$
C. $n, n + n$ and $n + n \times n$
D. $n, E + n$ and $E \times n$

gate2005 compiler-design grammar normal

2.6.26 Grammar: GATE2006-32, ISRO2016-35

https://gateoverflow.in/995

Consider the following statements about the context free grammar

$$G = \{S \rightarrow SS, S \rightarrow ab, S \rightarrow ba, S \rightarrow \epsilon\}$$

I. G is ambiguous
II. G produces all strings with equal number of a's and b's
III. G can be accepted by a deterministic PDA.

Which combination below expresses all the true statements about G?

A. I only
B. I and III only
C. II and III only
D. I, II and III

gate2006 compiler-design grammar normal isro2016

2.6.27 Grammar: GATE2006-59

https://gateoverflow.in/1837

Consider the following translation scheme.

- $S \rightarrow ER$
- $R \rightarrow^* E\{print('*');\}R \mid \varepsilon$
- $E \rightarrow F + E\{print('+');\} \mid F$
- $F \rightarrow (S) \mid id\{print(id.value);\}$

Here *id* is a token that represents an integer and *id.value* represents the corresponding integer value. For an input '$2 * 3 + 4$', this translation scheme prints

A. $2 * 3 + 4$
B. $2 * + 3\ 4$
C. $2\ 3 * 4 +$
D. $2\ 3\ 4 + *$

gate2006 compiler-design grammar normal

2.6.28 Grammar: GATE2006-84

https://gateoverflow.in/1856

Which one of the following grammars generates the language $L = \{a^i b^j \mid i \neq j\}$?

A. $S \rightarrow AC \mid CB$
$C \rightarrow aCb \mid a \mid b$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow Bb \mid \varepsilon$

B. $S \rightarrow aS \mid Sb \mid a \mid b$

C. $S \rightarrow AC \mid CB$
$C \rightarrow aCb \mid \varepsilon$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow Bb \mid \varepsilon$

D. $S \rightarrow AC \mid CB$
$C \rightarrow aCb \mid \varepsilon$
$A \rightarrow aA \mid a$
$B \rightarrow Bb \mid b$

gate2006 compiler-design grammar normal theory-of-computation

2.6.29 Grammar: GATE2006-85

https://gateoverflow.in/79799

Find the grammar that generates the language $L = \{a^i b^j \mid i \neq j\}$. In that grammar what is the length of the derivation (number of steps starring from S) to generate the string $a^l b^m$ with $l \neq m$

A. $max(l,m)+2$ B. $l+m+2$ C. $l+m+3$ D. $max(l,m)+3$

gate2006 compiler-design grammar normal

2.6.30 Grammar: GATE2006-85

https://gateoverflow.in/79801

The grammar

- $S \rightarrow AC \mid CB$
- $C \rightarrow aCb \mid \varepsilon$
- $A \rightarrow aA \mid a$
- $B \rightarrow Bb \mid b$

generates the language $L = \{a^i b^j \mid i \neq j\}$. In that grammar what is the length of the derivation (number of steps starting from S) to generate the string $a^l b^m$ with $l \neq m$

A. $max(l,m)+2$ B. $l+m+2$ C. $l+m+3$ D. $max(l,m)+3$

gate2006 compiler-design grammar normal

2.6.31 Grammar: GATE2007-52

https://gateoverflow.in/1250

Consider the grammar with non-terminals $N = \{S, C, S_1\}$, terminals $T = \{a, b, i, t, e\}$, with S as the start symbol, and the following set of rules:

$S \rightarrow iCtSS_1 \mid a$

$S_1 \rightarrow eS \mid \epsilon$

$C \rightarrow b$

The grammar is NOT LL(1) because:

A. it is left recursive
B. it is right recursive
C. it is ambiguous
D. it is not context-free

gate2007 compiler-design grammar normal

2.6.32 Grammar: GATE2007-53

https://gateoverflow.in/1251

Consider the following two statements:

- P: Every regular grammar is LL(1)
- Q: Every regular set has a 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

gate2007 compiler-design grammar normal

2.6.33 Grammar: GATE2007-78

https://gateoverflow.in/1272

Consider the CFG with $\{S, A, B\}$ as the non-terminal alphabet, $\{a, b\}$ as the terminal alphabet, S as the start symbol and the following set of production rules:

$S \rightarrow aB \qquad S \rightarrow bA$
$B \rightarrow b \qquad A \rightarrow a$
$B \rightarrow bS \qquad A \rightarrow aS$
$B \rightarrow aBB \qquad S \rightarrow bAA$

Which of the following strings is generated by the grammar?

A. $aaaabb$ B. $aabbbb$ C. $aabbab$ D. $abbbba$

gate2007 compiler-design grammar normal

2.6.34 Grammar: GATE2007-79

https://gateoverflow.in/43512

Consider the CFG with $\{S, A, B\}$ as the non-terminal alphabet, $\{a, b\}$ as the terminal alphabet, S as the start symbol and the following set of production rules:

$S \rightarrow aB \qquad S \rightarrow bA$
$B \rightarrow b \qquad A \rightarrow a$
$B \rightarrow bS \qquad A \rightarrow aS$
$B \rightarrow aBB \qquad S \rightarrow bAA$

For the string $aabbab$, how many derivation trees are there?

A. 1 B. 2 C. 3 D. 4

gate2007 compiler-design grammar normal

2.6.35 Grammar: GATE2007-IT-9

https://gateoverflow.in/3442

Consider an ambiguous grammar G and its disambiguated version D. Let the language recognized by the two grammars be denoted by $L(G)$ and $L(D)$ respectively. Which one of the following is true?

A. $L(D) \subset L(G)$
B. $L(D) \supset L(G)$
C. $L(D) = L(G)$
D. $L(D)$ is empty

gate2007-it compiler-design grammar normal

2.6.36 Grammar: GATE2008-50

https://gateoverflow.in/395

Which of the following statements are true?

I. Every left-recursive grammar can be converted to a right-recursive grammar and vice-versa
II. All ϵ-productions can be removed from any context-free grammar by suitable transformations
III. The language generated by a context-free grammar all of whose productions are of the form $X \rightarrow w$ or $X \rightarrow wY$ (where, w is a string of terminals and Y is a non-terminal), is always regular
IV. The derivation trees of strings generated by a context-free grammar in Chomsky Normal Form are always binary trees

A. I, II, III and IV
B. II, III and IV only
C. I, III and IV only
D. I, II and IV only

gate2008 normal compiler-design grammar

2.6.37 Grammar: GATE2010-38

https://gateoverflow.in/2339

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)

gate2010 compiler-design grammar normal

2.6.38 Grammar: GATE2014-2-17

https://gateoverflow.in/1973

Consider the grammar defined by the following production rules, with two operators $*$ and $+$

- $S \rightarrow T * P$

- $T \rightarrow U \mid T * U$
- $P \rightarrow Q + P \mid Q$
- $Q \rightarrow Id$
- $U \rightarrow Id$

Which one of the following is TRUE?

A. $+$ is left associative, while $*$ is right associative
B. $+$ is right associative, while $*$ is left associative
C. Both $+$ and $*$ are right associative
D. Both $+$ and $*$ are left associative

gate2014-2 compiler-design grammar normal

2.6.39 Grammar: GATE2016-2-45 https://gateoverflow.in/39594

Which one of the following grammars is free from left recursion?

A. $S \rightarrow AB$, $A \rightarrow Aa \mid b$, $B \rightarrow c$

B. $S \rightarrow Ab \mid Bb \mid c$, $A \rightarrow Bd \mid \epsilon$, $B \rightarrow e$

C. $S \rightarrow Aa \mid B$, $A \rightarrow Bb \mid Sc \mid \epsilon$, $B \rightarrow d$

D. $S \rightarrow Aa \mid Bb \mid c$, $A \rightarrow Bd \mid \epsilon$, $B \rightarrow Ae \mid \epsilon$

gate2016-2 compiler-design grammar easy

2.6.40 Grammar: GATE2016-2-46 https://gateoverflow.in/39598

A student wrote two context-free grammars G1 and G2 for generating a single C-like array declaration. The dimension of the array is at least one. For example,

```
int a[10] [3];
```

The grammars use D as the start symbol, and use six terminal symbols **int ; id [ ] num.**

Grammar **G1**	Grammar **G2**
D $\rightarrow$ **int L;**	D $\rightarrow$ **int L;**
L $\rightarrow$ **id** [**E**	L $\rightarrow$ **id E**
E $\rightarrow$ **num**]	E $\rightarrow$ **E** [**num**]
E $\rightarrow$ **num**] [**E**	E $\rightarrow$ [**num**]

Which of the grammars correctly generate the declaration mentioned above?

A. Both **G1** and **G2** B. Only **G1** C. Only **G2** D. Neither **G1** nor **G2**

gate2016-2 compiler-design grammar normal

2.6.41 Grammar: GATE2019-43 https://gateoverflow.in/302805

Consider the augmented grammar given below:

- $S' \rightarrow S$
- $S \rightarrow \langle L \rangle \mid id$
- $L \rightarrow L, S \mid S$

Let $I_0 = \text{CLOSURE}(\{[S' \rightarrow \cdot S]\})$. The number of items in the set $\text{GOTO}(I_0, \langle)$ is_____

gate2019 numerical-answers compiler-design grammar

2.7 Infix Postfix (1)

2.7.1 Infix Postfix: GATE1989-4-ii https://gateoverflow.in/87881

Provide short answers to the following questions:

Compute the postfix equivalent of the following infix arithmetic expression

$a + b * c + d * e \uparrow f$

where $\uparrow$ represents exponentiation. Assume normal operator precedences.

gate1989 descriptive compiler-design infix-postfix intermediate-code

2.8 Intermediate Code (8)

2.8.1 Intermediate Code: GATE1988-2xvii

https://gateoverflow.in/94350

Construct a DAG for the following set of quadruples:

- E:=A+B
- F:=E-C
- G:=F*D
- H:=A+B
- I:=I-C
- J:=I+G

gate1988 descriptive compiler-design intermediate-code

2.8.2 Intermediate Code: GATE1989-4-v

https://gateoverflow.in/87885

Is the following code template for the if-then-else statement correct? if not, correct it.

if expression then statement 1

else statement 2

Template:

Code for expression

(*result in $E, E > O$ indicates true *)

Branch on $E > O$ to $L1$

Code for statement 1

$L1$: Code for statement 2

descriptive gate1989 compiler-design intermediate-code

2.8.3 Intermediate Code: GATE1992-11b

https://gateoverflow.in/43583

Write 3 address intermediate code (quadruples) for the following boolean expression in the sequence as it would be generated by a compiler. Partial evaluation of boolean expressions is not permitted. Assume the usual rules of precedence of the operators.

$$(a + b) > (c + d) \text{ or } a > c \text{ and } b < d$$

gate1992 compiler-design syntax-directed-translation intermediate-code descriptive

2.8.4 Intermediate Code: GATE1994-1.12

https://gateoverflow.in/2453

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

gate1994 compiler-design intermediate-code easy

2.8.5 Intermediate Code: GATE2014-2-34

https://gateoverflow.in/1993

For a C program accessing $\mathbf{X[i][j][k]}$, 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.

```
t0 = i * 1024
t1 = j * 32
t2 = k * 4
t3 = t1 + t0
t4 = t3 + t2
t5 = X[t4]
```

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

A. $\mathbf{X}$ is declared as **"int $\mathbf{X[32][32][8]}$ "**.
B. $\mathbf{X}$ is declared as **"int $\mathbf{X[4][1024][32]}$ "**.
C. $\mathbf{X}$ is declared as **"char $\mathbf{X[4][32][8]}$ "**.
D. $\mathbf{X}$ is declared as **"char $\mathbf{X[32][16][2]}$ "**.

gate2014-2 compiler-design intermediate-code programming-in-c normal

2.8.6 Intermediate Code: GATE2014-3-17

https://gateoverflow.in/2051

One of the purposes of using intermediate code in compilers is to

A. make parsing and semantic analysis simpler.
B. improve error recovery and error reporting.
C. increase the chances of reusing the machine-independent code optimizer in other compilers.
D. improve the register allocation.

gate2014-3 compiler-design intermediate-code easy

2.8.7 Intermediate Code: GATE2015-1-55

https://gateoverflow.in/8365

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_____.

gate2015-1 compiler-design intermediate-code normal numerical-answers

2.8.8 Intermediate Code: GATE2015-1-8

https://gateoverflow.in/8096

For computer based on three-address instruction formats, each address field can be used to specify which of the following:

(S1) A memory operand

(S2) A processor register

(S3) An implied accumulator register

A. Either $S1$ or $S2$
B. Either $S2$ or $S3$
C. Only $S2$ and $S3$
D. All of $S1$, $S2$ and $S3$

gate2015-1 compiler-design intermediate-code normal

2.9 Left Recursion (1)

2.9.1 Left Recursion: GATE2017-2-32

https://gateoverflow.in/118374

Consider the following expression grammar G:

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

Which of the following grammars is not left recursive, but is equivalent to G?

A. $E \rightarrow E - T \mid T$ B. $E \rightarrow TE'$ C. $E \rightarrow TX$ D. $E \rightarrow TX \mid (TX)$

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

$E' \rightarrow -TE' \mid \epsilon$
$T \rightarrow T + F \mid F$
$F \rightarrow (E) \mid id$

$X \rightarrow -TX \mid \epsilon$
$T \rightarrow FY$
$Y \rightarrow +FY \mid \epsilon$
$F \rightarrow (E) \mid id$

$X \rightarrow -TX \mid +TX \mid \epsilon$
$T \rightarrow id$

gate2017-2 grammar left-recursion

2.10 Lexical Analysis (6)

2.10.1 Lexical Analysis: GATE1987-1-xvii

https://gateoverflow.in/80364

Using longer identifiers in a program will necessarily lead to:

A. Somewhat slower compilation
B. A program that is easier to understand
C. An incorrect program
D. None of the above

gate1987 compiler-design lexical-analysis

2.10.2 Lexical Analysis: GATE2000-1.18, ISRO2015-25

https://gateoverflow.in/641

The number of tokens in the following C statement is

```
printf("i=%d, &i=%x", i, &i);
```

A. 3 B. 26 C. 10 D. 21

gate2000 compiler-design lexical-analysis easy isro2015

2.10.3 Lexical Analysis: GATE2010-13

https://gateoverflow.in/2186

Which data structure in a compiler is used for managing information about variables and their attributes?

A. Abstract syntax tree
B. Symbol table
C. Semantic stack
D. Parse table

gate2010 compiler-design lexical-analysis easy

2.10.4 Lexical Analysis: GATE2011-1

https://gateoverflow.in/2103

In a compiler, keywords of a language are recognized during

A. parsing of the program
B. the code generation
C. the lexical analysis of the program
D. dataflow analysis

gate2011 compiler-design lexical-analysis easy

2.10.5 Lexical Analysis: GATE2011-19

https://gateoverflow.in/2121

The lexical analysis for a modern computer language such as Java needs the power of which one of the following machine models in a necessary and sufficient sense?

A. Finite state automata
B. Deterministic pushdown automata
C. Non-deterministic pushdown automata
D. Turing machine

gate2011 compiler-design lexical-analysis easy

2.10.6 Lexical Analysis: GATE2018-37

https://gateoverflow.in/204111

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: $a?(b \mid c)^*a$

T_2: $b?(a \mid c)^*b$

T_3: $c?(b \mid a)^*c$

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 processes 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

gate2018 compiler-design lexical-analysis normal

2.11 Linking (3)

2.11.1 Linking: GATE1991-03,ix

https://gateoverflow.in/519

Choose the correct alternatives (more than one may be correct) and write the corresponding letters only

A “link editor” is a program that:

A. matches the parameters of the macro-definition with locations of the parameters of the macro call
B. matches external names of one program with their location in other programs
C. matches the parameters of subroutine definition with the location of parameters of subroutine call.
D. acts as a link between text editor and the user
E. acts as a link between compiler and the user program

gate1991 compiler-design normal linking

2.11.2 Linking: GATE2003-76

https://gateoverflow.in/962

Which of the following is NOT an advantage of using shared, dynamically linked libraries as opposed to using statistically linked libraries?

A. Smaller sizes of executable files
B. Lesser overall page fault rate in the system
C. Faster program startup
D. Existing programs need not be re-linked to take advantage of newer versions of libraries

gate2003 compiler-design runtime-environments linking easy

2.11.3 Linking: GATE2004-9

https://gateoverflow.in/1006

Consider a program P that consists of two source modules M_1 and M_2 contained in two different files. If M_1 contains a reference to a function defined in M_2 the reference will be resolved at

A. Edit time B. Compile time C. Link time D. Load time

gate2004 compiler-design easy linking

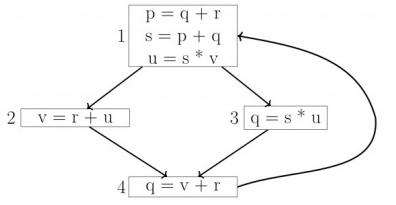
2.12 Live Variable (1)

2.12.1 Live Variable: GATE2015-1-50

https://gateoverflow.in/8356

A variable x is said to be live at a statement s_i in a program if the following three conditions hold simultaneously:

i. There exists a statement S_j that uses x
ii. There is a path from S_i to S_j in the flow graph corresponding to the program
iii. The path has no intervening assignment to x including at S_i and S_j



The variables which are live both at the statement in basic block 2 and at the statement in basic block 3 of the above control flow graph are

A. p, s, u B. r, s, u C. r, u D. q, v

gate2015-1 compiler-design live-variable normal

2.13 Macros (4)

2.13.1 Macros: GATE1992-01,vii

https://gateoverflow.in/552

Macro expansion is done in pass one instead of pass two in a two pass macro assembler because _____

gate1992 compiler-design macros easy

2.13.2 Macros: GATE1995-1.11

https://gateoverflow.in/2598

What are x and y in the following macro definition?

```
macro Add x, y
        Load y
        Mul x
        Store y
end macro
```

A. Variables
B. Identifiers
C. Actual parameters
D. Formal parameters

gate1995 compiler-design macros easy

2.13.3 Macros: GATE1996-2.16

https://gateoverflow.in/2745

Which of the following macros can put a macro assembler into an infinite loop?

i.
```
.MACRO M1, X
.IF EQ, X   ;if X=0 then
M1 X + 1
.ENDC
.IF NE, X   ;if X ≠ 0 then
.WORD X  ;address (X) is stored here
.ENDC
.ENDM
```

ii.
```
.MACRO M2, X
.IF EQ, X
M2 X
.ENDC
.IF NE, X
.WORD X + 1
.ENDC
.ENDM
```

A. (ii) only
B. (i) only
C. both (i) and (ii)
D. None of the above

gate1996 compiler-design macros normal

2.13.4 Macros: GATE1997-1.9

https://gateoverflow.in/2225

The conditional expansion facility of macro processor is provided to

A. test a condition during the execution of the expanded program
B. to expand certain model statements depending upon the value of a condition during the execution of the expanded program
C. to implement recursion
D. to expand certain model statements depending upon the value of a condition during the process of macro expansion

gate1997 compiler-design macros easy

2.14 Parameter Passing (13)

2.14.1 Parameter Passing: GATE1988-2xv

https://gateoverflow.in/94333

What is printed by following program, assuming call-by reference method of passing parameters for all variables in the parameter list of procedure P?

```
program     Main(inout, output);
var     a, b:integer;
        procedure P(x, y, z:integer);
        begin
            y:=y+1
            z:=x+x
        end P;
begin
    a:=2; b:=3;
```

```
    p(a+b, a, a);
    Write(a)
end.
```

gate1988 descriptive compiler-design runtime-environments parameter-passing numerical-answers

2.14.2 Parameter Passing: GATE1988-8i

https://gateoverflow.in/94371

Consider the procedure declaration:

```
Procedure
P (k: integer)
```

where the parameter passing mechanism is call-by-value-result. Is it correct if the call, P (A[i]), where A is an array and i an integer, is implemented as below.

a. create a new local variable, say z;
b. assign to z, the value of A [i];
c. execute the body of P using z for k;
d. set A [i] to z;

Explain your answer. If this is incorrect implementation, suggest a correct one.

gate1988 descriptive compiler-design runtime-environments parameter-passing

2.14.3 Parameter Passing: GATE1989-3-viii

https://gateoverflow.in/37264

In which of the following case(s) is it possible to obtain different results for call-by-reference and call-by-name parameter passing?

A. Passing an expression as a parameter
B. Passing an array as a parameter
C. Passing a pointer as a parameter
D. Passing as array element as a parameter

gate1989 parameter-passing runtime-environments compiler-design

2.14.4 Parameter Passing: GATE1990-11a

https://gateoverflow.in/85981

What does the following program output?

```
program module (input, output);
var
    a:array [1...5] of integer;
    i, j: integer;
procedure unknown (var b: integer, var c: integer);
var
    i:integer;
begin
    for i := 1 to 5 do a[i] := i;
    b:= 0; c := 0
    for i := 1 to 5 do write (a[i]);
    writeln();
    a[3]:=11; a[1]:=11;
    for i:=1 to 5 do a [i] := sqr(i);
    writeln(c,b); b := 5; c := 6;
end;
begin
  i:=1; j:=3; unknown (a[i], a[j]);
  for i:=1 to 5 do write (a[i]);
end;
```

gate1990 descriptive compiler-design runtime-environments parameter-passing

2.14.5 Parameter Passing: GATE1991-03,x

https://gateoverflow.in/524

Choose the correct alternatives (more than one may be correct) and write the corresponding letters only:

Indicate all the true statements from the following:

A. Recursive descent parsing cannot be used for grammar with left recursion.
B. The intermediate form for representing expressions which is best suited for code optimization is the postfix form.
C. A programming language not supporting either recursion or pointer type does not need the support of dynamic memory allocation.
D. Although C does not support call-by-name parameter passing, the effect can be correctly simulated in C
E. No feature of Pascal typing violates strong typing in Pascal.

gate1991 compiler-design parameter-passing programming difficult

2.14.6 Parameter Passing: GATE1991-09a

https://gateoverflow.in/536

Consider the following pseudo-code (all data items are of type integer):

```
procedure P(a, b, c);
    a := 2;
    c := a + b;
end {P}

begin
    x := 1;
    y := 5;
    z := 100;
    P(x, x*y, z);
    Write ('x = ', x, 'z = ', z);
end
```

Determine its output, if the parameters are passed to the Procedure P by

i. value
ii. reference
iii. name

gate1991 compiler-design parameter-passing normal runtime-environments

2.14.7 Parameter Passing: GATE1991-09b

https://gateoverflow.in/43603

For the following code, indicate the output if

a. static scope rules
b. dynamic scope rules

are used

```
var a,b : integer;

procedure P;
    a := 5;
    b := 10;
end {P};

procedure Q;
    var a, b : integer;
    P;
end {Q};

begin
    a := 1;
    b := 2;
    Q;
    Write ('a = ', a, 'b = ', b);
end
```

gate1991 runtime-environments normal compiler-design parameter-passing

2.14.8 Parameter Passing: GATE1993-26

https://gateoverflow.in/2322

A stack is used to pass parameters to procedures in a procedure call.

A. If a procedure P has two parameters as described in procedure definition:

```
procedure P (var x :integer; y: integer);
```

and if P is called by ; $P(a,b)$

State precisely in a sentence what is pushed on stack for parameters a and b

B. In the generated code for the body of procedure P, how will the addressing of formal parameters x and y differ?

gate1993 compiler-design parameter-passing runtime-environments normal

2.14.9 Parameter Passing: GATE1995-2.4

https://gateoverflow.in/2616

What is the value of X printed by the following program?

```
program COMPUTE (input, output);
var X:integer;
procedure FIND (X:real);
    begin
        X:=sqrt(X);
    end;
begin
    X:=2
    FIND(X);
    writeln(X);
end.
```

A. 2 B. $\sqrt{2}$ C. Run time error D. None of the above

gate1995 programming parameter-passing runtime-environments easy

2.14.10 Parameter Passing: GATE2003-74

https://gateoverflow.in/43575

The following program fragment is written in a programming language that allows global variables and does not allow nested declarations of functions.

```
global int i=100, j=5;
void P(x) {
 int i=10;
    print(x+10);
    i=200;
    j=20;
    print (x);
}
main() {P(i+j);}
```

If the programming language uses dynamic scoping and call by name parameter passing mechanism, the values printed by the above program are

A. $115, 220$ B. $25, 220$ C. $25, 15$ D. $115, 105$

gate2003 programming compiler-design parameter-passing runtime-environments normal

2.14.11 Parameter Passing: GATE2004-2,ISRO2017-54

https://gateoverflow.in/999

Consider the following function

```
void swap(int a, int b)
{
    int temp;
    temp = a;
    a = b;
    b = temp;
}
```

In order to exchange the values of two variables x and y.

A. call $swap(x, y)$
B. call $swap(\&x, \&y)$
C. $swap(x, y)$ cannot be used as it does not return any value
D. $swap(x, y)$ cannot be used as the parameters are passed by value

gate2004 compiler-design programming-in-c parameter-passing easy isro2017 runtime-environments

2.14.12 Parameter Passing: GATE2007-IT-33

https://gateoverflow.in/3466

Consider the program below in a hypothetical language which allows global variable and a choice of call by reference or call by value methods of parameter passing.

```
int i ;
program main ()
{
    int j = 60;
    i = 50;
    call f (i, j);
```

```
    print i, j;
}
procedure f (x, y)
{
    i = 100;
    x = 10;
    y = y + i ;
}
```

Which one of the following options represents the correct output of the program for the two parameter passing mechanisms?

A. Call by value : $i = 70, j = 10$; Call by reference :$i = 60, j = 70$
B. Call by value : $i = 50, j = 60$; Call by reference :$i = 50, j = 70$
C. Call by value : $i = 10, j = 70$; Call by reference :$i = 100, j = 60$
D. Call by value : $i = 100, j = 60$; Call by reference :$i = 10, j = 70$

gate2007-it programming parameter-passing normal compiler-design runtime-environments

2.14.13 Parameter Passing: GATE2016-1-36

https://gateoverflow.in/39701

What will be the output of the following pseudo-code when parameters are passed by reference and dynamic scoping is assumed?

```
a = 3;
void n(x) { x = x * a; print (x); }
void m(y) { a = 1 ; a = y - a;  n(a); print (a); }
void main () { m(a); }
```

A. $6, 2$ B. $6, 6$ C. $4, 2$ D. $4, 4$

gate2016-1 parameter-passing normal

2.15 Parsing (48)

2.15.1 Parsing: GATE1987-1-xiv

https://gateoverflow.in/80295

An operator precedence parser is a

A. Bottom-up parser.
B. Top-down parser.
C. Back tracking parser.
D. None of the above.

gate1987 compiler-design parsing

2.15.2 Parsing: GATE1988-10ia

https://gateoverflow.in/94390

Consider the following grammar:

- $S \rightarrow S$
- $S \rightarrow SS \mid a \mid \epsilon$

Construct the collection of sets of LR (0) items for this grammar and draw its goto graph.

gate1988 descriptive grammar parsing

2.15.3 Parsing: GATE1989-1-iii

https://gateoverflow.in/87046

Merging states with a common core may produce _____ conflicts and does not produce _____ conflicts in an LALR purser.

gate1989 descriptive compiler-design parsing

2.15.4 Parsing: GATE1992-02,xiii

https://gateoverflow.in/570

Choose the correct alternatives (more than one may be correct) and write the corresponding letters only:

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 define two sets LFOLLOW(A) and RFOLLOW(A) by replacing the word "sentential" by "left sentential" and "right most sentential" respectively in the definition of FOLLOW (A).

A. FOLLOW(A) and LFOLLOW(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.

gate1992 parsing compiler-design normal

2.15.5 Parsing: GATE1992-02,xiv

https://gateoverflow.in/571

Choose the correct alternatives (more than one may be correct) and write the corresponding letters only:

Consider the $SLR(1)$ and $LALR(1)$ parsing tables for a context free grammar. Which of the following statement is/are true?

A. The *goto* 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 tables may be different

gate1992 compiler-design normal parsing

2.15.6 Parsing: GATE1993-25

https://gateoverflow.in/2321

A simple Pascal like language has only three statements.

i. assignment statement e.g. x:=expression
ii. loop construct e.g. for i:=expression to expression do statement
iii. sequencing e.g. begin statement ;…; statement end

A. Write a context-free grammar (CFG) for statements in the above language. Assume that expression has already been defined. Do not use optional parenthesis and * operator in CFG.
B. Show the parse tree for the following statements:

```
for j:=2 to 10 do
begin
    x:=expr1;
    y:=expr2;
end
```

gate1993 compiler-design parsing normal

2.15.7 Parsing: GATE1995-8

https://gateoverflow.in/2643

Construct the LL(1) table for the following grammar.

1. $Expr \rightarrow \_Expr$
2. $Expr \rightarrow (Expr)$
3. $Expr \rightarrow Var\ ExprTail$
4. $ExprTail \rightarrow \_Expr$
5. $Expr \rightarrow \lambda$
6. $Var \rightarrow Id\ VarTail$
7. $VarTail \rightarrow (Expr)$
8. $VarTail \rightarrow \lambda$
9. $Goal \rightarrow Expr\ \$$

gate1995 compiler-design parsing normal

2.15.8 Parsing: GATE1998-1.26

https://gateoverflow.in/1663

Which of the following statements is true?

A. SLR parser is more powerful than LALR
B. LALR parser is more powerful than Canonical LR parser
C. Canonical LR parser is more powerful than LALR parser
D. The parsers SLR, Canonical CR, and LALR have the same power

gate1998 compiler-design parsing normal

2.15.9 Parsing: GATE1998-1.27

https://gateoverflow.in/1664

Type checking is normally done during

A. lexical analysis
B. syntax analysis
C. syntax directed translation
D. code optimization

gate1998 compiler-design parsing easy

2.15.10 Parsing: GATE1998-22

https://gateoverflow.in/1737

A. An identifier in a programming language consists of up to six letters and digits of which the first character must be a letter. Derive a regular expression for the identifier.
B. Build an $LL(1)$ parsing table for the language defined by the $LL(1)$ grammar with productions

$\text{Program} \rightarrow \text{ begin } d \text{ semi } X \text{ end}$
$X \rightarrow d \text{ semi } X \mid sY$
$Y \rightarrow \text{ semi } sY \mid \epsilon$

gate1998 compiler-design parsing descriptive

2.15.11 Parsing: GATE1999-1.17

https://gateoverflow.in/1470

Which of the following is the most powerful parsing method?

A. LL (1) B. Canonical LR C. SLR D. LALR

gate1999 compiler-design parsing easy

2.15.12 Parsing: GATE2000-1.19, UGCNET-Dec2013-II-30

https://gateoverflow.in/642

Which of the following derivations does a top-down parser use while parsing an input string? The input is assumed to be scanned in left to right order.

A. Leftmost derivation
B. Leftmost derivation traced out in reverse
C. Rightmost derivation
D. Rightmost derivation traced out in reverse

gate2000 compiler-design parsing normal ugcnetdec2013ii

2.15.13 Parsing: GATE2001-16

https://gateoverflow.in/757

Consider the following grammar with terminal alphabet $\Sigma = \{a, (,), +, *\}$ and start symbol E. The production rules of the grammar are:

- $E \rightarrow aA$
- $E \rightarrow (E)$
- $A \rightarrow +E$
- $A \rightarrow *E$
- $A \rightarrow \epsilon$

a. Compute the FIRST and FOLLOW sets for E and A.
b. Complete the LL(1) parse table for the grammar.

gate2001 compiler-design parsing normal

2.15.14 Parsing: GATE2002-22

https://gateoverflow.in/875

A. Construct all the parse trees corresponding to $i + j * k$ for the grammar
$E \rightarrow E + E$
$E \rightarrow E * E$
$E \rightarrow id$
B. In this grammar, what is the precedence of the two operators $*$ and $+$?

C. If only one parse tree is desired for any string in the same language, what changes are to be made so that the resulting LALR(1) grammar is unambiguous?

gate2002 compiler-design parsing normal descriptive

2.15.15 Parsing: GATE2003-16

https://gateoverflow.in/906

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

gate2003 compiler-design parsing easy

2.15.16 Parsing: GATE2003-17

https://gateoverflow.in/907

Assume that the SLR parser for a grammar G has n_1 states and the LALR parser for G has n_2 states. The relationship between n_1 and n_2 is

A. n_1 is necessarily less than n_2
B. n_1 is necessarily equal to n_2
C. n_1 is necessarily greater than n_2
D. None of the above

gate2003 compiler-design parsing easy

2.15.17 Parsing: GATE2003-57

https://gateoverflow.in/945

Consider the grammar shown below.

$S \rightarrow C\,C$

$C \rightarrow c\,C \mid d$

This grammar is

A. LL(1)
B. SLR(1) but not LL(1)
C. LALR(1) but not SLR(1)
D. LR(I) but not LALR(1)

gate2003 compiler-design grammar parsing normal

2.15.18 Parsing: GATE2005-14

https://gateoverflow.in/1350

The grammar $A \rightarrow AA \mid (A) \mid \epsilon$ is not suitable for predictive-parsing because the grammar is:

A. ambiguous B. left-recursive C. right-recursive D. an operator-grammar

gate2005 compiler-design parsing grammar easy

2.15.19 Parsing: GATE2005-60

https://gateoverflow.in/1383

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. $n_1 \geq n_3 \geq n_2$

gate2005 compiler-design parsing normal

2.15.20 Parsing: GATE2005-83a

https://gateoverflow.in/1405

Statement for Linked Answer Questions 83a & 83b:

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

$$\begin{array}{l|l} E \rightarrow number & E.val = number.val \\ \mid E \text{ `+' } E & E^{(1)}.val = E^{(2)}.val + E^{(3)}.val \\ \mid E \text{ `}\times\text{' } E & E^{(1)}.val = E^{(2)}.val \times E^{(3)}.val \end{array}$$

The above grammar and the semantic rules are fed to a *yaac* 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 *yaac* 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

gate2005 compiler-design parsing difficult

2.15.21 Parsing: GATE2005-83b

https://gateoverflow.in/87037

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

$$\begin{array}{l|l} E \rightarrow number & E.val = number.val \\ \mid E \text{ `+' } E & E^{(1)}.val = E^{(2)}.val + E^{(3)}.val \\ \mid E \text{ `}\times\text{' } E & E^{(1)}.val = E^{(2)}.val \times E^{(3)}.val \end{array}$$

Assume the conflicts of this question are resolved using yacc tool 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 '$\times$' is higher than that of '$+$', and both operators are left associative; expression is evaluated to 7
D. Precedence of '$+$' is higher than that of '$\times$', and both operators are left associative; expression is evaluated to 9

gate2005 compiler-design parsing normal

2.15.22 Parsing: GATE2005-IT-83a

https://gateoverflow.in/3849

Consider the context-free grammar

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

where E is the starting symbol, the set of terminals is $\{id, (, +,), *\}$, and the set of nonterminals is $\{E\}$.

Which of the following terminal strings has more than one parse tree when parsed according to the above grammar?

A. $id + id + id + id$
B. $id + (id * (id * id))$
C. $(id * (id * id)) + id$
D. $((id * id + id) * id)$

gate2005-it compiler-design grammar parsing easy

2.15.23 Parsing: GATE2005-IT-83b

https://gateoverflow.in/3850

Consider the context-free grammar

- $E \rightarrow E + E$
- $E \rightarrow (E * E)$
- $E \rightarrow \text{id}$

where E is the starting symbol, the set of terminals is $\{id, (, +,), *\}$, and the set of non-terminals is $\{E\}$.

For the terminal string $id + id + id + id$, how many parse trees are possible?

A. 5 B. 4 C. 3 D. 2

gate2005-it compiler-design parsing normal

2.15.24 Parsing: GATE2006-58

https://gateoverflow.in/1836

Consider the following grammar:

$S \rightarrow FR$

$R \rightarrow *S \mid \varepsilon$

$F \rightarrow id$

In the predictive parser table, M, of the grammar the entries M[S,id] and M[R,$] respectively are

A. $\{S \rightarrow FR\}$ and $\{R \rightarrow \varepsilon\}$
B. $\{S \rightarrow FR\}$ and $\{\}$
C. $\{S \rightarrow FR\}$ and $\{R \rightarrow *S\}$
D. $\{F \rightarrow id\}$ and $\{R \rightarrow \varepsilon\}$

gate2006 compiler-design parsing normal

2.15.25 Parsing: GATE2006-7

https://gateoverflow.in/886

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 the above

gate2006 compiler-design parsing normal

2.15.26 Parsing: GATE2007-18

https://gateoverflow.in/1216

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.

gate2007 compiler-design parsing normal

2.15.27 Parsing: GATE2008-11

https://gateoverflow.in/409

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 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

gate2008 compiler-design parsing normal

2.15.28 Parsing: GATE2008-55

https://gateoverflow.in/478

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

gate2008 compiler-design parsing normal

2.15.29 Parsing: GATE2008-IT-79

https://gateoverflow.in/3393

A CFG G is given with the following productions where S is the start symbol, A is a non-terminal and a and b are terminals.

- $S \rightarrow aS \mid A$
- $A \rightarrow aAb \mid bAa \mid \epsilon$

For the string "$aabbaab$" how many steps are required to derive the string and how many parse trees are there?

A. 6 and 1 B. 6 and 2 C. 7 and 2 D. 4 and 2

gate2008-it compiler-design context-free-language parsing normal

2.15.30 Parsing: GATE2009-42

https://gateoverflow.in/1328

Which of the following statements are TRUE?

I. There exist parsing algorithms for some programming languages whose complexities are less than $\Theta(n^3)$
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

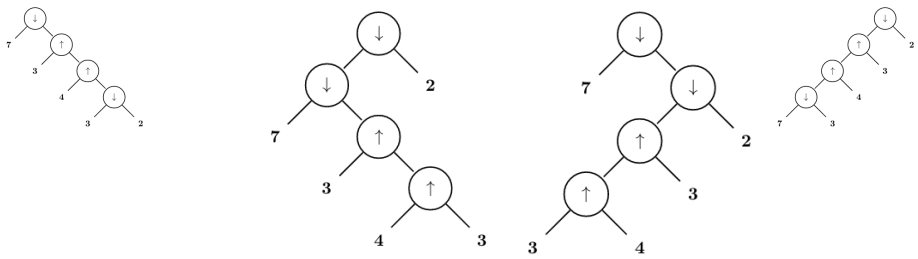
gate2009 compiler-design parsing normal

2.15.31 Parsing: GATE2011-27

https://gateoverflow.in/2129

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

A. B. C. D.



gate2011 compiler-design parsing normal

2.15.32 Parsing: GATE2012-52

https://gateoverflow.in/2181

For the grammar below, a partial $LL(1)$ parsing table is also presented along with the grammar. Entries that need to be filled are indicated as $E1, E2,$ and $E3$. ε is the empty string, \$ indicates end of input, and, $\mid$ separates alternate right hand sides of productions.

- $S \rightarrow aAbB \mid bAaB \mid \varepsilon$
- $A \rightarrow S$
- $B \rightarrow S$

	a	b	$
S	E1	E2	$S \rightarrow \varepsilon$
A	$A \rightarrow S$	$A \rightarrow S$	error
B	$B \rightarrow S$	$B \rightarrow S$	$E3$

The FIRST and FOLLOW sets for the non-terminals A and B are

A. $\text{FIRST}(A) = \{a, b, \varepsilon\} = \text{FIRST}(B)$
$\text{FOLLOW}(A) = \{a, b\}$
$\text{FOLLOW}(B) = \{a, b, \$\}$

B. $\text{FIRST}(A) = \{a, b, \$\}$
$\text{FIRST}(B) = \{a, b, \varepsilon\}$
$\text{FOLLOW}(A) = \{a, b\}$
$\text{FOLLOW}(B) = \{\$\}$

C. $\text{FIRST}(A) = \{a, b, \varepsilon\} = \text{FIRST}(B)$
$\text{FOLLOW}(A) = \{a, b\}$
$\text{FOLLOW}(B) = \varnothing$

D. $\text{FIRST}(A) = \{a, b\} = \text{FIRST}(B)$
$\text{FOLLOW}(A) = \{a, b\}$
$\text{FOLLOW}(B) = \{a, b\}$

gate2012 compiler-design parsing normal

2.15.33 Parsing: GATE2012-53

https://gateoverflow.in/43312

For the grammar below, a partial $LL(1)$ parsing table is also presented along with the grammar. Entries that need to be filled are indicated as $E1, E2,$ and $E3$. ε is the empty string, $ indicates end of input, and, $\mid$ separates alternate right hand sides of productions.

- $S \rightarrow aAbB \mid bAaB \mid \varepsilon$
- $A \rightarrow S$
- $B \rightarrow S$

	a	b	$
S	E1	E2	$S \rightarrow \varepsilon$
A	$A \rightarrow S$	$A \rightarrow S$	error
B	$B \rightarrow S$	$B \rightarrow S$	$E3$

The appropriate entries for $E1, E2,$ and $E3$ are

A. $E1 : S \rightarrow aAbB, A \rightarrow S$
$E2 : S \rightarrow bAaB, B \rightarrow S$
$E1 : B \rightarrow S$

B. $E1 : S \rightarrow aAbB, S \rightarrow \varepsilon$
$E2 : S \rightarrow bAaB, S \rightarrow \varepsilon$
$E3 : S \rightarrow \varepsilon$

C. $E1 : S \rightarrow aAbB, S \rightarrow \varepsilon$
$E2 : S \rightarrow bAaB, S \rightarrow \varepsilon$
$E3 : B \rightarrow S$

D. $E1 : A \rightarrow S, S \rightarrow \varepsilon$
$E2 : B \rightarrow S, S \rightarrow \varepsilon$
$E3 : B \rightarrow S$

normal gate2012 compiler-design parsing

2.15.34 Parsing: GATE2013-40

https://gateoverflow.in/1551

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

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

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

1. Cannot be merged since look aheads 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

gate2013 compiler-design parsing normal

2.15.35 Parsing: GATE2013-9

https://gateoverflow.in/1418

What is the maximum number of reduce moves that can be taken by a bottom-up parser for a grammar with no epsilon and unit-production (i.e., of type $A \rightarrow \epsilon$ and $A \rightarrow a$) to parse a string with n tokens?

A. $n/2$ B. $n-1$ C. $2n-1$ D. 2^n

gate2013 compiler-design parsing normal

2.15.36 Parsing: GATE2014-1-34

https://gateoverflow.in/1807

A canonical set of items is given below

$S \rightarrow L. > R$

$Q \rightarrow R.$

On input symbol $<$ the set has

A. a shift-reduce conflict and a reduce-reduce conflict.
B. a shift-reduce conflict but not a reduce-reduce conflict.
C. a reduce-reduce conflict but not a shift-reduce conflict.
D. neither a shift-reduce nor a reduce-reduce conflict.

gate2014-1 compiler-design parsing normal

2.15.37 Parsing: GATE2015-3-16

https://gateoverflow.in/8413

Among simple LR (SLR), canonical LR, and look-ahead LR (LALR), which of the following pairs identify the method that is very easy to implement and the method that is the most powerful, in that order?

A. SLR, LALR
B. Canonical LR, LALR
C. SLR, canonical LR
D. LALR, canonical LR

gate2015-3 compiler-design parsing normal

2.15.38 Parsing: GATE2015-3-31

https://gateoverflow.in/8488

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?

S1: LL(1) can parse all strings that are generated using grammar G

S2: LR(1) can parse all strings that are generated using grammar G

A. Only S1 B. Only S2 C. Both S1 and S2 D. Neither S1 and S2

gate2015-3 compiler-design parsing normal

2.15.39 Parsing: GATE2016-1-45

https://gateoverflow.in/39697

The attribute of three arithmetic operators in some programming language are given below.

OPERATOR	PRECEDENCE	ASSOCIATIVITY	ARITY
$+$	High	Left	Binary
$-$	Medium	Right	Binary
$*$	Low	Left	Binary

The value of the expression $2-5+1-7*3$ in this language is _____.

gate2016-1 compiler-design parsing normal numerical-answers

2.15.40 Parsing: GATE2017-1-17

https://gateoverflow.in/118297

Consider the following grammar:

- $P \rightarrow xQRS$
- $Q \rightarrow yz \mid z$
- $R \rightarrow w \mid \varepsilon$
- $S \rightarrow y$

What is FOLLOW(Q)?

A. $\{R\}$ B. $\{w\}$ C. $\{w, y\}$ D. $\{w, \$\}$

gate2017-1 compiler-design parsing

2.15.41 Parsing: GATE2017-1-43

https://gateoverflow.in/118326

Consider the following grammar:

- stmt $\rightarrow$ **if** expr **then** expr **else** expr; stmt | 0
- expr $\rightarrow$ term **relop** term | term
- term $\rightarrow$ id | number
- id $\rightarrow$ **a** | **b** | **c**
- number $\rightarrow [0-9]$

where **relop** is a relational operator (e.g.. $<$. $>$ ...). 0 refers to the empty statement, and **if**, **then**, **else** are terminals.
Consider a program P following the above grammar containing ten **if** terminals. The number of control flow paths in P is_____ . For example. the program
if e_1 **then** e_2 **else** e_3
has 2 control flow paths. $e_1 \rightarrow e_2$ and $e_1 \rightarrow e_3$.

gate2017-1 compiler-design parsing normal numerical-answers

2.15.42 Parsing: GATE2017-2-6

https://gateoverflow.in/118343

Which of the following statements about parser is/are CORRECT?

I. Canonical LR is more powerful than SLR
II. SLR is more powerful than LALR
III. SLR is more powerful than Canonical LR

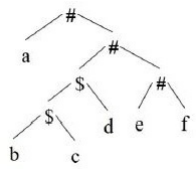
A. I only B. II only C. III only D. II and III only

gate2017-2 compiler-design parsing

2.15.43 Parsing: GATE2018-38

https://gateoverflow.in/204112

Consider the following parse tree for the expression a#b cd#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

gate2018 compiler-design parsing normal

2.15.44 Parsing: GATE2019-19

https://gateoverflow.in/302829

Consider the grammar given below:

- $S \rightarrow Aa$
- $A \rightarrow BD$
- $B \rightarrow b \mid \epsilon$
- $D \rightarrow d \mid \epsilon$

Let a, b, d and $\$$ be indexed as follows:

a	b	d	$\$$
3	2	1	0

Compute the FOLLOW set of the non-terminal B and write the index values for the symbols in the FOLLOW set in the descending order.(For example, if the FOLLOW set is $(a, b, d, \$)$, then the answer should be 3210)

gate2019 numerical-answers compiler-design parsing

2.15.45 Parsing: GATE2019-3

https://gateoverflow.in/302845

Which one of the following kinds of derivation is used by LR parsers?

A. Leftmost
B. Leftmost in reverse
C. Rightmost
D. Rightmost in reverse

gate2019 compiler-design parsing

2.15.46 Parsing: TIFR2012-B-17

https://gateoverflow.in/25215

Which of the following correctly describes $LR(k)$ parsing?

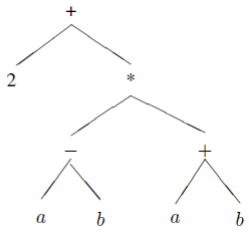
A. The input string is alternately scanned left to right and right to left with k reversals.
B. Input string is scanned once left to right with rightmost derivation and k symbol look-ahead.
C. $LR(k)$ grammers are expressively as powerful as context-free grammers.
D. Parser makes k left-to-right passes over input string.
E. Input string is scanned from left to right once with k symbol to the right as look-ahead to give left-most derivation.

tifr2012 compiler-design parsing

2.15.47 Parsing: TIFR2012-B-8

https://gateoverflow.in/25108

Consider the parse tree



Assume that $*$ has higher precedence than $+$, $-$ and operators associate right to left (i.e $(a+b+c=(a+(b+c))$. Consider

i. $2+a-b$
ii. $2+a-b*a+b$
iii. $2+((a-b)*(a+b)))$
iv. $2+(a-b)*(a+b)$

The parse tree corresponds to

A. Expression (i)
B. Expression (ii)
C. Expression (iv) only
D. Expression (ii), (iii), and (iv)
E. Expression (iii) and (iv) only

tifr2012 compiler-design parsing

2.15.48 Parsing: TIFR2015-B-15

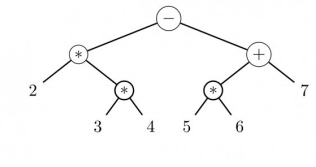
https://gateoverflow.in/30079

Consider the following grammar (the start symbol is E) for generating expressions.

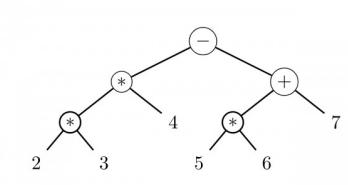
- $E \rightarrow T - E \mid T + E \mid T$
- $T \rightarrow T * F \mid F$
- $F \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

With respect to this grammar, which of the following trees is the valid evaluation tree for the expression $2 * 3 * 4 - 5 * 6 + 7$?

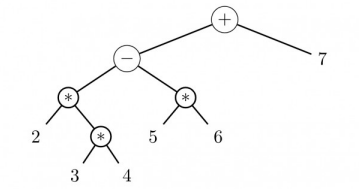
a.



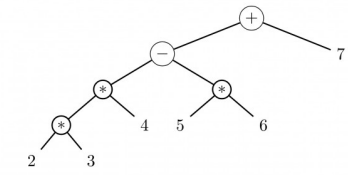
b.



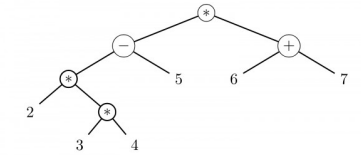
c.



d.



e.



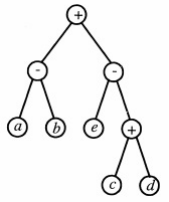
tifr2015 parsing

2.16 Register Allocation (2)

2.16.1 Register Allocation: GATE2011-36

https://gateoverflow.in/2138

Consider evaluating the following expression tree on a machine with load-store architecture in which memory can be accessed only through load and store instructions. The variables $a, b, c, d,$ and e are initially stored in memory. The binary operators used in this expression tree can be evaluated by the machine only when operands are in registers. The instructions produce result only in a register. If no intermediate results can be stored in memory, what is the minimum number of registers needed to evaluate this expression?



A. 2 B. 9 C. 5 D. 3

gate2011 compiler-design register-allocation normal

2.16.2 Register Allocation: GATE2017-1-52

https://gateoverflow.in/118746

Consider the expression $(a-1) * (((b+c)/3)+d)$. Let X be the minimum number of registers required by an *optimal* code generation (without any register spill) algorithm for a load/store architecture, in which

A. only load and store instructions can have memory operands and
B. arithmetic instructions can have only register or immediate operands. The value of X is _____ .

gate2017-1 compiler-design register-allocation normal numerical-answers

2.17 Runtime Environments (18)

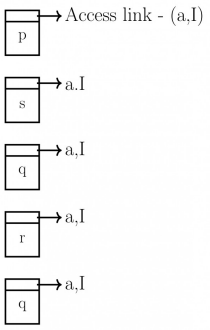
2.17.1 Runtime Environments: GATE1988-2xii

https://gateoverflow.in/93966

Consider the following program skeleton and below figure which shows activation records of procedures involved in the calling sequence.

$$p \rightarrow s \rightarrow q \rightarrow r \rightarrow q.$$

Write the access links of the activation records to enable correct access and variables in the procedures from other procedures involved in the calling sequence




```
procedure p;
  procedure q;
    procedure r;
      begin
        q
      end r;
    begin
    r
    end q;
    procedure s;
      begin
        q
      end s;
  begin
    s
  end p;
```

gate1988 normal descriptive runtime-environments compiler-design

2.17.2 Runtime Environments: GATE1989-10a

https://gateoverflow.in/89636

Will recursion work correctly in a language with static allocation of all variables? Explain.

gate1989 descriptive compiler-design runtime-environments

2.17.3 Runtime Environments: GATE1989-8b

https://gateoverflow.in/89082

Indicate the result of the following program if the language uses (i) static scope rules and (ii) dynamic scope rules.

```
        var x, y:integer;
        procedure A (var z:integer);
        var x:integer;
        begin x:=1; B; z:= x end;
        procedure B;
        begin x:=x+1 end;
        begin
        x:=5; A(y); write (y)
...end.
```

gate1989 descriptive compiler-design runtime-environments

2.17.4 Runtime Environments: GATE1990-2-v

https://gateoverflow.in/83980

Match the pairs in the following questions:

(a)	Pointer data type	(p)	Type conversion
(b)	Activation record	(q)	Dynamic data structure
(c)	Repeat-until	(r)	Recursion
(d)	Coercion	(s)	Nondeterministic loop

gate1990 match-the-following compiler-design runtime-environments recursion

2.17.5 Runtime Environments: GATE1990-4-v

https://gateoverflow.in/85394

State whether the following statements are TRUE or FALSE with reason:

The Link-load-and-go loading scheme required less storage space than the link-and-go loading scheme.

gate1990 true-false compiler-design runtime-environments

2.17.6 Runtime Environments: GATE1993-7.7

https://gateoverflow.in/2295

A part of the system software which under all circumstances must reside in the main memory is:

A. text editor B. assembler C. linker D. loader E. none of the above

gate1993 compiler-design runtime-environments easy

2.17.7 Runtime Environments: GATE1995-1.14

https://gateoverflow.in/2601

A linker is given object modules for a set of programs that were compiled separately. What information need to be included in an object module?

A. Object code
B. Relocation bits
C. Names and locations of all external symbols defined in the object module
D. Absolute addresses of internal symbols

gate1995 compiler-design runtime-environments normal

2.17.8 Runtime Environments: GATE1997-1.10

https://gateoverflow.in/2226

Heap allocation is required for languages.

A. that support recursion
B. that support dynamic data structure
C. that use dynamic scope rules
D. None of the above

gate1997 compiler-design easy runtime-environments

2.17.9 Runtime Environments: GATE1997-1.8

https://gateoverflow.in/2224

A language L allows declaration of arrays whose sizes are not known during compilation. It is required to make efficient use of memory. Which one of the following is true?

A. A compiler using static memory allocation can be written for L
B. A compiler cannot be written for L; an interpreter must be used
C. A compiler using dynamic memory allocation can be written for L
D. None of the above

gate1997 compiler-design easy runtime-environments

2.17.10 Runtime Environments: GATE1998-1.25, ISRO2008-41

https://gateoverflow.in/1662

In a resident – OS computer, which of the following systems must reside in the main memory under all situations?

A. Assembler B. Linker C. Loader D. Compiler

gate1998 compiler-design runtime-environments normal isro2008

2.17.11 Runtime Environments: GATE1998-1.28

https://gateoverflow.in/1665

A linker reads four modules whose lengths are $200, 800, 600$ and 500 words, respectively. If they are loaded in that order, what are the relocation constants?

A. $0, 200, 500, 600$
B. $0, 200, 1000, 1600$
C. $200, 500, 600, 800$
D. $200, 700, 1300, 2100$

gate1998 compiler-design runtime-environments normal

2.17.12 Runtime Environments: GATE1998-2.15

https://gateoverflow.in/1687

Faster access to non-local variables is achieved using an array of pointers to activation records called a

A. stack
B. heap
C. display
D. activation tree

gate1998 programming compiler-design normal runtime-environments

2.17.13 Runtime Environments: GATE2001-1.17

https://gateoverflow.in/710

The process of assigning load addresses to the various parts of the program and adjusting the code and the data in the program to reflect the assigned addresses is called

A. Assembly
B. parsing
C. Relocation
D. Symbol resolution

gate2001 compiler-design runtime-environments easy

2.17.14 Runtime Environments: GATE2008-54

https://gateoverflow.in/477

Which of the following are true?

I. 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
II. Multi-level access link (or display) arrangement is needed to arrange activation records only if the programming language being implemented has nesting of procedures/functions
III. Recursion in programming languages cannot be implemented with dynamic storage allocation
IV. Nesting procedures/functions and recursion require a dynamic heap allocation scheme and cannot be implemented with a stack-based allocation scheme for activation records
V. 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. II and V only
B. I, III and IV only
C. I, II and V only
D. II, III and V only

gate2008 compiler-design difficult runtime-environments

2.17.15 Runtime Environments: GATE2010-14

https://gateoverflow.in/2187

Which languages necessarily need heap allocation in the runtime environment?

A. Those that support recursion.
B. Those that use dynamic scoping.
C. Those that allow dynamic data structure.
D. Those that use global variables.

gate2010 compiler-design easy runtime-environments

2.17.16 Runtime Environments: GATE2012-36

https://gateoverflow.in/1758

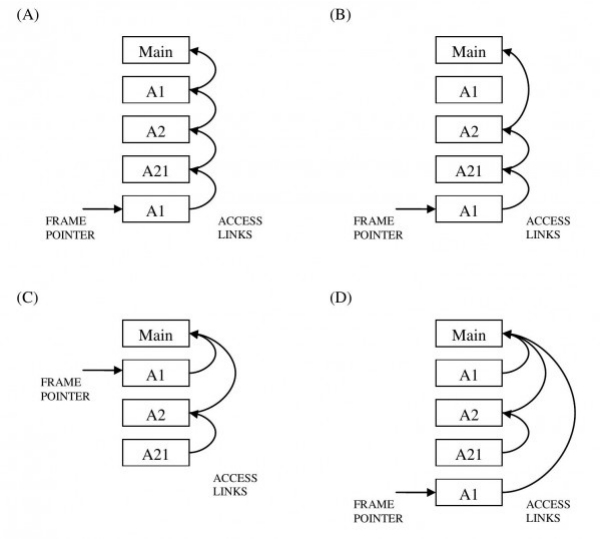
Consider the program given below, in a block-structured pseudo-language with lexical scoping and nesting of procedures permitted.

```
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: $Main \rightarrow A1 \rightarrow A2 \rightarrow A21 \rightarrow A1$

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



gate2012 compiler-design runtime-environments normal

2.17.17 Runtime Environments: GATE2014-2-18

https://gateoverflow.in/1975

Which one of the following is **NOT** performed during compilation?

A. Dynamic memory allocation
B. Type checking
C. Symbol table management
D. Inline expansion

gate2014-2 compiler-design easy runtime-environments

2.17.18 Runtime Environments: GATE2014-3-18

https://gateoverflow.in/2052

Which of the following statements are CORRECT?

1. Static allocation of all data areas by a compiler makes it impossible to implement recursion.
2. Automatic garbage collection is essential to implement recursion.
3. Dynamic allocation of activation records is essential to implement recursion.
4. Both heap and stack are essential to implement recursion.

A. 1 and 2 only B. 2 and 3 only C. 3 and 4 only D. 1 and 3 only

gate2014-3 compiler-design runtime-environments normal

2.18 Static Single Assignment (2)

2.18.1 Static Single Assignment: GATE2016-1-19

https://gateoverflow.in/39675

Consider the following code segment.

```
x = u - t;
y = x * v;
x = y + w;
y = t - z;
y = x * y;
```

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

_____.

gate2016-1 compiler-design static-single-assignment normal numerical-answers

2.18.2 Static Single Assignment: GATE2017-1-12

https://gateoverflow.in/118292

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.
```
p1 = a  - b
q1 = p1 * c
p1 = u  * v
q1 = p1 + q1
```

B.
```
p3 = a  - b
q4 = p3 * c
p4 = u  * v
q5 = p4 + q4
```

C.
```
p1 = a  - b
q1 = p2 * c
p3 = u  * v
q2 = p4 + q3
```

D.
```
p1 = a - b
q1 = p * c
p2 = u * v
q2 = p + q
```

gate2017-1 compiler-design intermediate-code normal static-single-assignment

2.19 Syntax Directed Translation (9)

2.19.1 Syntax Directed Translation: GATE1992-11a

https://gateoverflow.in/590

Write syntax directed definitions (semantic rules) for the following grammar to add the type of each identifier to its entry in the symbol table during semantic analysis. Rewriting the grammar is not permitted and semantic rules are to be added to the ends of productions only.

- $D \rightarrow TL;$
- $T \rightarrow \text{int}$
- $T \rightarrow \text{real}$
- $L \rightarrow L,id$
- $L \rightarrow id$

gate1992 compiler-design syntax-directed-translation normal

2.19.2 Syntax Directed Translation: GATE1995-2.10

https://gateoverflow.in/2622

A shift reduce parser carries out the actions specified within braces immediately after reducing with the corresponding rule of grammar

- $S \rightarrow xxW$ $\{\text{print}\text{“1”}\}$
- $S \rightarrow y$ $\{\text{print}\text{“2”}\}$
- $W \rightarrow Sz$ $\{\text{print}\text{“3”}\}$

What is the translation of $xxxyzz$ using the syntax directed translation scheme described by the above rules?

A. 23131 B. 11233 C. 11231 D. 33211

gate1995 compiler-design grammar syntax-directed-translation normal

2.19.3 Syntax Directed Translation: GATE1996-20

https://gateoverflow.in/2772

Consider the syntax-directed translation schema (SDTS) shown below:

- $E \rightarrow E + E$ {print “+”}
- $E \rightarrow E * E$ {print “.”}
- $E \rightarrow id$ {print id.name}
- $E \rightarrow (E)$

An LR-parser executes the actions associated with the productions immediately after a reduction by the corresponding production. Draw the parse tree and write the translation for the sentence.

$(a+b)*(c+d)$, using SDTS given above.

gate1996 compiler-design syntax-directed-translation normal

2.19.4 Syntax Directed Translation: GATE1998-23

https://gateoverflow.in/1738

Let the attribute 'val' give the value of a binary number generated by S in the following grammar:

- $S \rightarrow L.L \mid L$
- $L \rightarrow LB \mid B$
- $B \rightarrow 0 \mid 1$

For example, an input 101.101 gives $S.val = 5.625$

Construct a syntax directed translation scheme using only synthesized attributes, to determine $S.val$.

gate1998 compiler-design syntax-directed-translation normal descriptive

2.19.5 Syntax Directed Translation: GATE2000-19

https://gateoverflow.in/690

Consider the syntax directed translation scheme (SDTS) given in the following. Assume attribute evaluation with bottom-up parsing, i.e., attributes are evaluated immediately after a reduction.

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

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

T$\rightarrow$ F - T_1 {T.val = F.val - T_1.val}

T $\rightarrow$ F {T.val = F.val}

F $\rightarrow$2 {F.val = 2}

F $\rightarrow$4 {F.val = 4}

A. Using this SDTS, construct a parse tree for the expression $4-2-4*2$ and also compute its $E.val$.
B. It is required to compute the total number of reductions performed to parse a given input. Using synthesized attributes only, modify the SDTS given, without changing the grammar, to find $E.red$, the number of reductions performed while reducing an input to E.

gate2000 compiler-design syntax-directed-translation normal descriptive

2.19.6 Syntax Directed Translation: GATE2001-17

https://gateoverflow.in/758

The syntax of the repeat-until statement is given by the following grammar

$S \rightarrow$ repeat S_1 until E

where E stands for expressions, S and S_1 stand for statements. The non-terminals S and S_1 have an attribute code that represents generated code. The non-terminal E has two attributes. The attribute code represents generated code to evaluate the expression and store its value in a distinct variable, and the attribute varName contains the name of the variable in which the truth value is stored. The truth-value stored in the variable is 1 if E is true, 0 if E is false.

Give a syntax-directed definition to generate three-address code for the repeat-until statement. Assume that you can call a function newlabel() that returns a distinct label for a statement. Use the operator '\\' to concatenate two strings and the function gen(s) to generate a line containing the string s.

gate2001 compiler-design syntax-directed-translation normal descriptive

2.19.7 Syntax Directed Translation: GATE2003-18

https://gateoverflow.in/908

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

gate2003 compiler-design syntax-directed-translation normal

2.19.8 Syntax Directed Translation: GATE2016-1-46

https://gateoverflow.in/39700

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

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

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

gate2016-1 compiler-design syntax-directed-translation normal

2.19.9 Syntax Directed Translation: GATE2019-36

https://gateoverflow.in/302812

Consider the following grammar and the semantic actions to support that inherited type declaration attributes. Let X_1, X_2, X_3, X_4, X_5, and X_6 be the placeholders for the non-terminals D, T, L or L_1 in the following table:

Production rule	Semantic action
$D \rightarrow TL$	$X_1.\text{type} = X_2.\text{type}$
$T \rightarrow \text{int}$	$T.\text{type} = \text{int}$
$T \rightarrow \text{float}$	$T.\text{type} = \text{float}$
$L \rightarrow L_1, id$	$X_3.\text{type} = X_4.\text{type}$ $\text{addType}(id.\text{entry}, X_5.\text{type})$
$L \rightarrow id$	$\text{addType}(id.\text{entry}, X_6, \text{type})$

Which one of the following are appropriate choices for X_1, X_2, X_3 and X_4?

A. $X_1 = L,\ X_2 = T,\ X_3 = L_1,\ X_4 = L$
B. $X_1 = T,\ X_2 = L,\ X_3 = L_1,\ X_4 = T$
C. $X_1 = L,\ X_2 = L,\ X_3 = L_1,\ X_4 = T$
D. $X_1 = T,\ X_2 = L,\ X_3 = T,\ X_4 = L_1$

gate2019 compiler-design syntax-directed-translation

2.20 Target Code Generation (4)

2.20.1 Target Code Generation: GATE1997-4.9

https://gateoverflow.in/2250

The expression $(a * b) * c\ op \ldots$

where 'op' is one of '$+$', '$*$' and '$\uparrow$' (exponentiation) can be evaluated on a CPU with single register without storing the value of $(a * b)$ if

A. 'op' is '$+$' or '$*$'
B. 'op' is '$\uparrow$' or '$*$'
C. 'op' is '$\uparrow$' or '$+$'
D. not possible to evaluate without storing

gate1997 compiler-design target-code-generation register-allocation normal

2.20.2 Target Code Generation: GATE2003-59

https://gateoverflow.in/947

Consider the syntax directed definition shown below.

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

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 *ti*'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. $t_1 = Y; t_2 = Z; t_3 = t_1 + t_2; X = t_3$

gate2003 compiler-design target-code-generation normal

2.20.3 Target Code Generation: GATE2004-10

https://gateoverflow.in/4069

Consider the grammar rule $E \rightarrow E1 - E2$ for arithmetic expressions. The code generated is targeted to a CPU having a single user register. The subtraction operation requires the first operand to be in the register. If $E1$ and $E2$ do not have any common sub expression, in order to get the shortest possible code

A. $E1$ should be evaluated first
B. $E2$ should be evaluated first
C. Evaluation of $E1$ and $E2$ should necessarily be interleaved
D. Order of evaluation of $E1$ and $E2$ is of no consequence

gate2004 compiler-design target-code-generation normal

2.20.4 Target Code Generation: GATE2010-37

https://gateoverflow.in/2338

The program below uses six temporary variables a, b, c, d, e, f.

```
a = 1
b = 10
c = 20
d = a + b
e = c + d
f = c + e
b = c + e
e = b + f
d = 5 + e
return d + f
```

Assuming that all operations take their operands from registers, what is the minimum number of registers needed to execute this program without spilling?

A. 2 B. 3 C. 4 D. 6

gate2010 compiler-design target-code-generation register-allocation normal

2.21 Variable Scope (2)

2.21.1 Variable Scope: GATE1987-1-xix

https://gateoverflow.in/80373

Study the following program written in a block-structured language:

```
Var x, y:interger;
procedure P(n:interger);
begin
     x:=(n+2)/(n-3);
end;

procedure Q
Var x, y:interger;
begin
    x:=3;
    y:=4;
    P(y);
    Write(x)                                  __(1)
end;

begin
    x:=7;
    y:=8;
    Q;
Write(x);                                     __(2)
end.
```

What will be printed by the write statements marked (1) and (2) in the program if the variables are statically scoped?

A. 3,6 B. 6,7 C. 3,7 D. None of the above.

gate1987 compiler-design variable-scope

2.21.2 Variable Scope: GATE1987-1-xx

https://gateoverflow.in/80374

For the program given below what will be printed by the write statements marked (1) and (2) in the program if the variables are dynamically scoped?

```
Var x, y:interger;
procedure P(n:interger);
begin
     x := (n+2)/(n-3);
end;

procedure Q
Var x, y:interger;
begin
    x:=3;
    y:=4;
    P(y);
    Write(x);                                  __(1)
end;

begin
    x:=7;
    y:=8;
    Q;
    Write(x);                                    __(2)
end.
```

A. $3,6$ B. $6,7$ C. $3,7$ D. None of the above

gate1987 compiler-design variable-scope

2.22 Viable Prefix (1)

2.22.1 Viable Prefix: GATE2015-1-13

https://gateoverflow.in/8187

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

gate2015-1 compiler-design parsing normal viable-prefix