GATE CS Topic wise Questions

Compiler Design

YEAR 2001

Question. 1

Which of the following statements is false ?

(A) An unambiguous grammar has same left most and right most 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

SOLUTION

Yes, the $LL(1)$ parser is top down parser.
Order of strength $LR < SLR < LALR$.
So (A) & (C) are, true.
An ambiguous grammar can't be $LR(K)$
So option (A) is false since an unambiguous grammar has unique right most derivation & left most derivations but both are not same.
Hence (A) is correct option

YEAR 2002

Question. 2

Dynamic linking can cause security concerns because

(A) Security is dynamic

(B) The path for searching dynamic libraries is not known till run

time.

(C) Linking is insecure

(D) Cryptographic procedures are not available for dynamic linking

SOLUTION

Dynamic linking is type of linking in which libraries required by the program are linked during run time. But at this time cryptographic procedures are not available, so make this process insecure.
Hence (D) is correct option.

YEAR 2003

Question. 3

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 this

SOLUTION

If a grammar has left recursion & left factoring then it is ambiguous. So to convert a CFG to $LL(1)$ grammar both removal of left recursion & left factoring need to be done.
Hence (C) is correct option.

Question. 4

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

SOLUTION

SLR parsue is less range of context free languages than *LALR* but still both n_1 & n_2 are same for *SLR* & *LALR* respectively.
Hence (B) is correct option.

Question. 5

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

(A) always be evaluated

(B) be evaluated if the definition is L-attributed

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

(D) never be evaluated

SOLUTION

Every *S* (synthesized) -attributed definitions is *L*- attributed. So in a bottom-up evaluation of *SDD* inherited attributes can be evaluated only if the definition has synthesized attributes.
Hence (C) is correct option.

Question. 6

Which of the following statements is FALSE?

(A) In statically typed language, each variable in a program has a fixed type

(B) In up-typed languages, values do not have any types

(C) In dynamically typed languages, variables have no types

(D) In all statically typed languages, each variable in a program is associated with values of only a single type during the execution of the program

SOLUTION

(1) True for statically typed languages where each variable has fixed type. Similarly (4) is also correct.

(2) True, in un-typed languages types of values are not defined.

But option (C) is false, since in dynamically typed language variables have dynamically changing types but not that they have no type.
Hence (C) is correct option.

Question. 7

Consider the grammar shown below

$$S \rightarrow | \; EtSS' \,|\, \alpha$$
$$S' \rightarrow eS \,|\, \in$$
$$E \rightarrow b$$

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

(A) $\{s' \rightarrow eS\}$ and $\{S' \rightarrow \varepsilon\}$ (B) $\{s' \rightarrow eS\}$ and $\{\}$

(C) $\{s' \rightarrow \varepsilon\}$ and $\{S' \rightarrow \varepsilon\}$ (D) $\{s' \rightarrow eS, S' \rightarrow \varepsilon\}$ and $\{ S' \rightarrow \varepsilon\}$

SOLUTION

Given grammar

$$S \rightarrow EtSS' \,|\, \alpha$$
$$S' \rightarrow eS \,|\, \varepsilon$$
$$E \rightarrow b$$

Terminals	i	t	a	e	b	\$
$N.T$						
S	$S \rightarrow iEtSS'$		$S \rightarrow \alpha$			
S'				$S' \rightarrow eS$		$S' \rightarrow \varepsilon$
				$S' \rightarrow \varepsilon$		
E					$E \rightarrow b$	

Predictive Parsing table.

So this table presents predictive parsing for dangling if else & shows ambiguity

$$M[S', e] = \{S' \rightarrow eS, S' \rightarrow \varepsilon\}$$
$$M[S', \$] = \{S' \rightarrow \varepsilon\}$$

Hence (D) is correct option.

Question. 8

Consider the grammar shown below.

$$S \rightarrow C \; C$$
$$C \rightarrow eC \,|\, d$$

The grammar is

(A) LL (1)

(B) SLR (1) but not LL (1)

(C) LALR (1) but not SLR (1)

(D) LR (1) but not LALR (1)

SOLUTION

Given grammar

$$S \rightarrow CC$$
$$C \rightarrow cC \mid d$$

it can't be LL since $C \rightarrow cC$ is recursive. $LR(1)$ also known as CLR parser, and every CF grammar is CLR grammar.

So (A) is false but (C) & (D) can be correct.

This grammar is CLR and also reducible to $LALR$ without any conflicts. So (D) is false.

Only need to check for $SLR(1)$ or $LR(0)$

This grammar is not SLR.

Hence (C) is correct option

Question. 9

Consider the translation scheme shown below

$S \rightarrow TR$

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

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

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

(A) $9+5+2$ (B) $95+2+$

(C) $952++$ (D) $++952$

SOLUTION

$$S \rightarrow TR$$
$$R \rightarrow + \ T\{print('+');\}R \mid \varepsilon$$
$$T \rightarrow \text{num}\{\text{print}(\text{num.val});\}$$

Given string $9+5+2$

$$S \rightarrow TR$$

$T + TR$	$\{\text{print}(+);\}$
$T + T + T$	$\{\text{print}(+);\}$
$9 + T + T$	$\{\text{print}(9);\}$
$9 + 5 + T$	$\{\text{print}(5);\}$
$9 + 5 + 2$	$\{\text{print}(2);\}$

So $++952$ is printed

Hence (D) is correct option.

Question. 10

Consider the syntax directed definition shown below

```
S → id:=E
  →                    newtemp ();
                    gen(t    . place    . place;);
                     .place   t}
  →                  .place     .place;}
```

Here, gen is a function that generates the output code, and newtemp is a function that returns the name of a new temporary variable on every call. Assume that t_1'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) $t_1 = Y; t_2 = Z; t_3 + t_2; X = t_3$

SOLUTION

In 3-address code we use temporary variables to reduce complex instructions so here

$$t_1 = Y$$
$$t_2 = Z$$
$$t_3 = t_1 + t_2$$
$$x = t_3$$

Hence (D) is correct option.

Data for Q. 11 & 12 are given below.

Solve the problems and choose the correct answers.

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

```
global inti
void
          int i
          print
          i

          print
}
main () { (i   ) }
```


Question. 11

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

(A) 115, 220 (B) 25, 220

(C) 25, 15 (D) 115, 105

SOLUTION

In static scoping the variables are initialized at compile time only

So $i = 100$ & $j = 5$

$$P(i+j) = P(100+5) = P(105)$$

So $x = 105$

$x + 10 = 105 + 10 = 115$

So 115 & 105 will be printed.

Hence (D) is correct option.

Question. 12

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

SOLUTION

In dynamic scoping, the local values are considered & variables are initialized at run time.

Since $x = i + j$ & in $P(x)$

$i = 200$ & $j = 20$

$x = 200 + 20 = 220$

& printing $(x + 10)$

$x = i + j + 10$

$= 10 + 5 + 10 = 25$

Hence (B) is correct option

Question. 13

Consider the following class definitions in a hypothetical object oriented language that supports inheritance and uses dynamic binding. The language should not be assumed to be either Java or C++, thought the syntax is similar

```
Class     P {        Class Q subclass of P {
     void f(int i) {     void f (int i) {
          print (i);          print (   i);
     }                   }
}                     }
```

Now consider the following program fragment:

$P\ x =$new $Q()$;

$Q\ y =$new $Q()$;

$P\ z =$new $Q()$;

$x.f(1);((P)y).f(1);z.f(1);$

Here $((P)y)$ denotes a typecast of y to P. The output produced by executing the above program fragment will be

(A) 1 2 1 (B) 2 1 1

(C) 2 1 2 (D) 2 2 2

SOLUTION

1. $Px = newQ();$
2. $Qy = newQ();$
3. $Pz = newQ();$
4. $x \cdot f(1)$; print $2 \times i = 2$
5. $((P)\, y) \cdot f(1);$
6. $z \cdot f(1)$ print $2 \times i = 2$

but line 5. will print 2 because typecast to parent class can't prevent over ridding. So function $f(1)$ of class Q will be called not $f(1)$ of class P.
Hence (D) is correct option.

Question. 14

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

(A) Smaller sizes of executable

(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

SOLUTION

The advantages of shared dynamically linked libraries include.
(A) smaller size of executable since less data

(B) lesser overall page fault rate.

(C) No need for re-linking if newer versions of libraries are there.

But since compilation time doesn't include linking so a long linking time required during runtime in *DLL's* so slow startup.
Hence (C) is correct option.

YEAR 2004

Question. 15

Which of the following grammar rules violate the requirements of an

operator grammar? P, Q, R are non-terminals, and r, s, t are terminals .

(i) $P \rightarrow QR$

(ii) $P \rightarrow Q\ s\ R$

(iii) $P \rightarrow \varepsilon$

(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

SOLUTION

(I) $P \rightarrow QR$ is not possible since two NT should include one operator as Terminal.

(II) Correct

(III) Again incorrect.

(IV) Correct.

Hence (B) is correct option.

Question. 16

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

SOLUTION

The two modules needed to be linked since definition exist & $M2$ & $M1$ refers it. So during linking phase $M1$ links to $M2$.

Hence (C) is correct option.

Question. 17

Consider the grammar rule $E \rightarrow E_1 - E_2$ 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 E_1 and E_2 do not have any common sub expression, in order to get the shortest possible code

(A) E_1 should be evaluated first

(B) E_2 should be evaluated first

(C) Evaluation of E_1 and E_2 should necessarily be interleaved

(D) Order of evaluation of E_1 and E_2 is of no consequence

SOLUTION

$E1$ is to be kept in accumulator & accumulator is required for operations to evaluate $E2$ also. So $E2$ should be evaluated first & then $E1$, so finally $E1$ will be in accumulator, otherwise need to use move & load instructions.
Hence (B) is correct option.

Question. 18

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

$E \rightarrow E_1 \# T$	value = .value * .value}
|	.value = .value}
→	.value = .value + .value}
	.value = .value}
→num	.value =num.value}

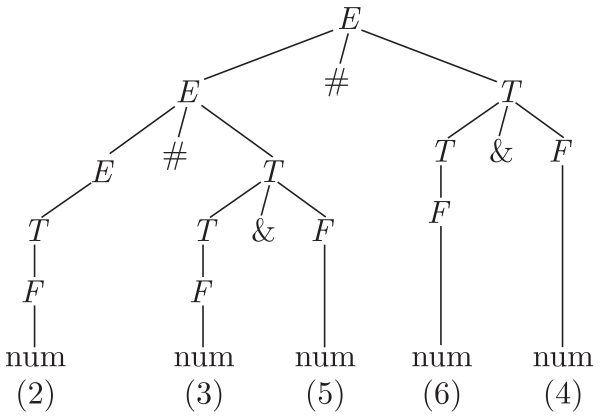
Compute E. value for the root of the parse tree for the expression: 2 # 3 # & 5 # 6 & 4.

(A) 200 (B) 180

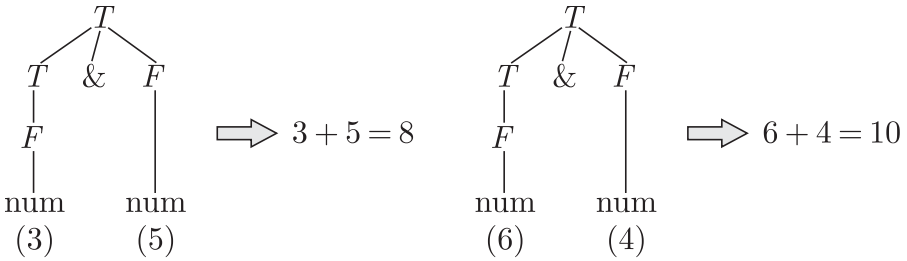
(C) 160 (D) 40

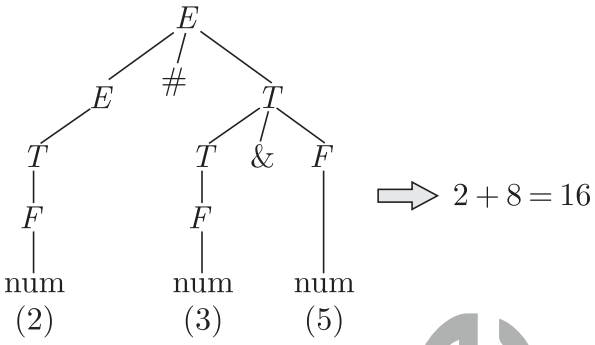
SOLUTION

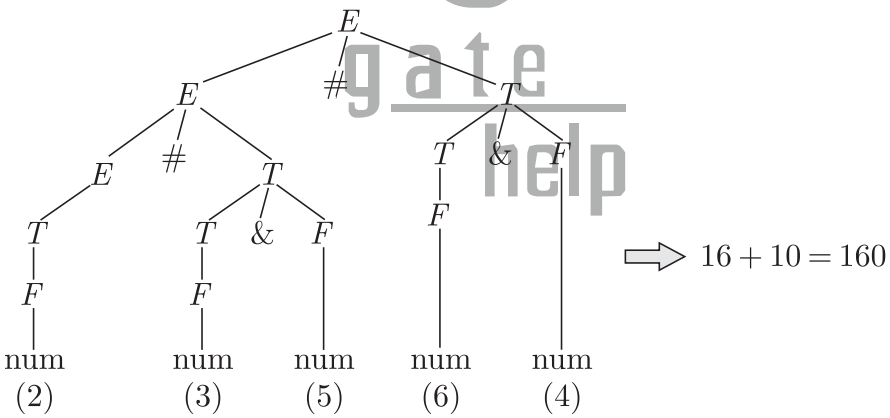
The parse tree would be.



Now we evaluate bottom up.







LL → left to right left most derivation no ambignity should be there

SLR or $LR(0)$ L to R reverse right sentential form create $LR(0)$ items.

CLR or $LR(1)$ create $LR(1)$ items no bound

$LALR$ reduced CLR if while reducing any conflict found then not $LALR$

Hence (C) is correct option.

YEAR 2005

Question. 19

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

(A) ambiguous
(B) Left-recursive
(C) right-recurisve
(D) an operator-grammar

SOLUTION

The grammar is definitely left & right recursive but it is not suitable for predictive parsing because it is ambiguous.
Hence (A) is correct option.

Question. 20

Consider the grammar $E \rightarrow E + n \mid E \times n \mid n$

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

(A) $n, E + n$ and $\mathrm{E} + \mathrm{n} \times \mathrm{n}$
(B) $n, E + n$ and $\mathrm{E} + \mathrm{E} \times \mathrm{n}$
(C) $n, n + n$ and $n + \mathrm{n} \times \mathrm{n}$
(D) $n, E + n$ and $\mathrm{E} \times \mathrm{n}$

SOLUTION

Given grammar

$$E \rightarrow E + n$$
$$E \rightarrow E \times n$$
$$E \rightarrow n$$
$$\text{String} = n + n \times n$$

Right sentential so right most non terminal will be used.

$$E \rightarrow E \times n \qquad \{E \rightarrow E \times n\}$$
$$E + n \times n \qquad \{E \rightarrow E + n\}$$
$$n + n \times n \qquad \{E \rightarrow n\}$$

So during reduction the order is reverse.
So $\{E \rightarrow n, E \rightarrow E + n, E \rightarrow E \times n\}$
Hence (D) is correct option.

Question. 21

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 $n_1 n_2$ 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) $n_1 \geq n_3 \geq n_2$

SOLUTION

The no. of states for $SLR(1)$ & $LALR(1)$ are equal so $n_1 = n_3$, but $CLR(1)$ or $LR(1)$ will have no. of states greater than $LALR$ & $LR(0)$ both.
Hence (B) is correct option.

Question. 22

Consider line number 3 of the following C-program.

```
int main (){                    | * Line 1 * |
int 1, N;                       | * line  2 * |
fro (i   ,1  N,1  );            | * 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

SOLUTION

There are no lexical errors for C because all the wrong spelled keywords would be considered as identifiers until the syntax is checked.
So the compiler would give syntax errors.
Hence (C) is correct option.

Data for Q. 23 & 24 are given below.

Solve the problems and choose the correct answers.

Consider the following expression grammar. The semantic rules for

expression calculation are stared next to each grammar production.

```
E →  number     Eval        number val
      E         E  .val     E  .VAL   E  .val
      E × E     E  .val     E  .VAL   E  .val
     ;
```

Question. 23

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

(A) It detects recursion and eliminates recursion

(B) It detects reduce-reduce conflict, and resolves

(C) It detects shift-reduce conflict, and resolves the conflict in favor of a shift over a reduce action

(D) It detects shift-reduce conflict, and resolves the conflict in favor of a reduce over a shift action

SOLUTION

Yace tool is used to create a $LALR(1)$ parser. This parser can detect the conflicts but to resolve the conflicts it actually prefers shift over reduce action.
Hence (C) is correct option.

Question. 24

Assume the conflicts part (a) of this question are resolved and an LALR(1) parser is generated for parsing arithmetic expressions as per the given grammar. Consider an expression $3 \times 2 + 1$. What precedence and associativity properties does the generated parser realize?

(A) Equal precedence and left associativity; expression is evaluated to 7

(B) Equal precedence and right associativity, expression is evaluated to 9

(C) Precedence of 'x' is higher than that of '+', and both operators are left associative; expression is evaluated to 7

(D) Precedence of '$\times$' is higher than that of '$\times$', and both operators are left associative; expression is evaluated to 9

SOLUTION

The grammar has equal precedence and it is also ambiguous. Since $LALR(1)$ parser prefer shift over reduce so $+$ operation will be executed here before $*$. $2+1=3$ & $3\times 3=9$ also the operators are right associative.
Hence (B) is correct option.

YEAR 2006

Question. 25

Consider the following grammar.

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

Consider the following $LR(0)$ items corresponding to the grammar above.

(i) $S \rightarrow S * .E$

(ii) $E \rightarrow F. + E$

(iii) $E \rightarrow F + .E$

Given the items above, which two of them will appear in the same set in the canonical sets-of-items for the grammar?

(A) (i) and (ii) (B) (ii) and (iii)

(C) (i) and (iii) (D) None of these

SOLUTION

If $S \rightarrow S * \cdot E$ is in $LR(0)$ then $E \rightarrow F + \cdot E$ will also be there because both of them has '$\cdot$' before E.
Hence (C) is correct option.

Question. 26

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

(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 $\{R \rightarrow \varepsilon\}$

SOLUTION

The predictive parser table is given as.

Non Terminal	$*$	id	$\$$
S		$S \rightarrow FR$	
F		$F \rightarrow id$	
R	$R \rightarrow * S$ $R \rightarrow \in$		$R \rightarrow \in$

So at $M[S, id] = \{S \rightarrow FR\}$
$M[R,\$] = \{R \rightarrow \in\}$

Hence (A) is correct option.

Question. 27

Consider the following translation scheme.

```
S → ER
R → * E{print{' * ');R | ε
E → F + E{print(' + '); | F
F → (S) | id{print(id.value);}
```

Here id is a taken 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 + *

SOLUTION

Input string $2 * 3 + 4$

$S \rightarrow ER$
FR
$idR \quad \{\text{print}(2)\}$
$id * ER \quad \{\text{print}(*)\}$
$id * F + ER \quad \{\text{print}(+)\}$
$id * id + ER \quad \{\text{print}(3)\}$
$id * id + FR$
$id * id + idR \quad \{\text{print } (4)\}$
$id * id + id$

So $2 * + 3\ 4$ are printed
Hence (B) is correct option.

Question. 28

Consider the following C code segment.

```
for
      for
            if  i
                        X i
                  }
            }
}
```

Which one to the following false?
(A) The code contains loop-in variant computation
(B) There is scope of common sub-expression elimination in this code
(C) There is scope strength reduction in this code
(D) There is scope of dead code elimination in this code

SOLUTION

All the statements are true except option (D) since there is no dead code to get eliminated.
Hence (D) is correct option.

Question. 29

Which one of the following grammars generates the language

$L = (a^i b^i \mid i \neq j\}$?

(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) $S \rightarrow ACCB$
$C \rightarrow aCb \mid \in$
$A \rightarrow aA \mid \in$
$B \rightarrow Bb \mid \in$

(D) $S \rightarrow AC \mid CB$
$C \rightarrow aCb \mid \in$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$

SOLUTION

The grammar

$$S \rightarrow AC \mid CB$$
$$C \rightarrow aCb \mid \in$$
$$A \rightarrow aA \mid a$$
$$B \rightarrow bB \mid b$$

Consider string $aaabb$

$$S \rightarrow AC$$
$$AaCb$$
$$AaaCbb$$
$$Aaabb$$
$$aaabb$$

But string $aabb$

$$S \rightarrow AC$$

And this string is not derivable.
Hence (D) is correct option.

Question. 30

In the correct grammar above, 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 $(l, m) + 3$

SOLUTION

It is very clear from the previous solution that the no. of steps required

depend upon the no. of $a's$ & $b's$ which ever is higher & exceeds by 2 due to $S \rightarrow AC \mid CB$ & $C \rightarrow \in$
So $\max(l, m) + 2$
Hence (A) is correct option.

YEAR 2007

Question. 31

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

SOLUTION

Clearly LR & $LALR$ are not top down they are bottom up passers.
Also not operator precedence parser.
But yes recursive descent parser is top down parser. Starts from start symbol & derives the terminal string.
Hence (A) is correct option.

Question. 32

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

$S \rightarrow iCtSS_1 \mid a$

$S_1 \rightarrow eS \mid \varepsilon$

$C \rightarrow b$

The grammar is NOTLL(1) because:

(A) It is left recursive (B) It is right recursive

(C) It is ambiguous (D) It is not context-free

SOLUTION

The grammar has production
$S \rightarrow iCtSS_1$ here the right hand side of grammar has the same symbol as left side. So the grammar is left recursive.
The grammar is not ambiguous.
Hence (A) is correct option.

Question. 33

Consider the following two statements:

P: Every regular grammar is LL(1)

Q: Every regular set has LR(1) grammar

Which of the following is TRUE?

(A) Both P and Q are true (B) P is true and Q is false

(C) P is false and Q is true (D) Both P and Q are false

SOLUTION

$LL(1)$ parsers can recognize the regular grammars also $LL(1)$ is subset of $LR(1)$ or CLR grammar so it also recognizes regular sets. So both accept regular grammar.

Question. 34

In a simplified computer the instructions are:

$OP\ R_j, R_i$ – Performs $R_j\ OP\ R_i$ and stores the result in register R_i

$OP\ m, R_i$ – Performs val OP R_i abd stores the result in R_i. value denotes the content of memory location m.

$MCVm, R_i$ –Moves the content off memory loction m to register R_i.

$MCVm, Ri, m$ –Moves the content of register R_i to memory location m.

The computer has only two registers, and OP is either ADD or SUB. Consider the following basic block:

$$t_1 = a + b$$
$$t_2 = c + d$$
$$t_3 = e - t_2$$
$$t_4 = t_1 - t_2$$

Assume that all operands are initially in memory. The final value of the computation should be in memory. What is the minimum number of MOV instructions in the code generated for this basic block?

(A) 2 (B) 3

(C) 5 (D) 6

SOLUTION

The operation sequence would be

$MOV\ \ a, R_1$
$ADD\ \ b, R_1\ \ \{R_1 = t_1$
$MOV\ \ c, R_2$
$ADD\ \ d, R_2\ \ \{R_2 = t_2$
$SUB\ \ e, R_2\ \ \{t_3 = e - R_2 = R_2$
$SUB\ \ R_1, R_2\ \ \{R_2 = t_4$
$MOV\ \ R_2, t_4\ \ \{$finally in memory

Totally no. of move operation are 3
Hence (B) is correct option

Data for Q. 35 & 36 are given below.

Solve the problems and choose the correct answers.

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

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

Question. 35

Which of the following strings is generated by the grammar?

(A) $aaaabb$ (B) $aabbbb$

(C) $aabbab$ (D) $abbbba$

SOLUTION

$$
\begin{aligned}
& aabbab \\
S \rightarrow\ & aB \\
\rightarrow\ & aaBB \\
\rightarrow\ & aabSB \\
\rightarrow\ & aabbAB \\
\rightarrow\ & aabbab
\end{aligned}
$$

Hence (C) is correct option.

Question. 36

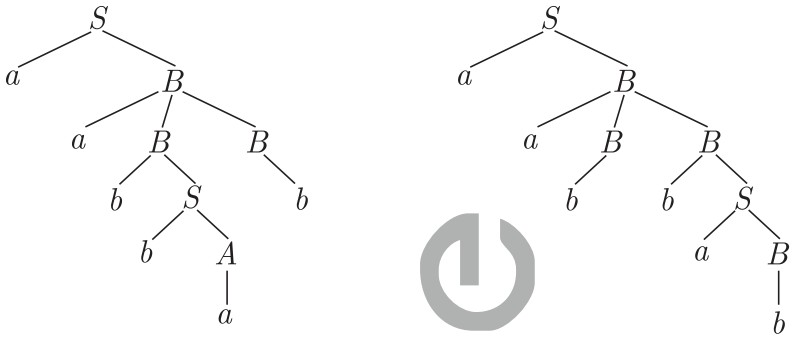
For the correct answer string to Q. 9 how many derivation trees are there?

(A) 1 (B) 2

(C) 3 (D) 4

SOLUTION

For the derivation two trees are possible



So due to ambiguity 2 trees are possible
Hence (B) is correct option.

YEAR 2008

Question. 37

Which of the following describes a handle (as applicable to LR-parsing) appropriately?

(A) It is the position in a sentential form where the next shift or reduce operation will occur

(B) It is a non-terminal whose production will be used for reduction in the next step

(C) It is a production that may be used for reduction in a future step along with a position in the sentential form where the next shift or reduce operation will occur.

(D) It is the production p that will be used for reduction in the next step along with a position in the sentential form where the right hand side of the production may be found

SOLUTION

Handles are the part of sentential form, & they are identified as the right side of any given production which will be used for reduction in the net step.
Hence (D) is correct option.

Question. 38

Some code optimizations are carried out on the intermediate code because

(A) They enhance the portability of the complier to other target processors

(B) Program analysis is name accurate on intermediate code than on machine code

(C) The information from data flow analysis cannot otherwise be used for optimization

(D) The information from the front end cannot otherwise be used for optimization

SOLUTION

Code optimizations are carried out on the intermediate code because program analysis is more accurate on intermediate code than on machine code.
Hence (B) is correct option.

Question. 39

Which of the following are true?

(i) A programming language option does not permit global variables of any king 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/function

(iii) Recursion in programming languages cannot be implemented with dynamic storage allocation

(iv) Nesting of procedures/functions and recursion require a dynamic heap allocation scheme and cannot be implemented with a stack-based allocation scheme for activation records

(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) (D) (ii), (iii) and (v) only

SOLUTION

I. Statement is false since global variables are required for recursions with static storage. This is due to unavailability of stack in static storage.
II. This is true
III. In dynamic allocation heap structure is used, so it is false.
IV. False since recursion can be implemented.
V. Statement is completely true.

So only II & V are true.
Hence (A) is correct option.

Question. 40

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

SOLUTION

$LALR$ parser is reduced form of CLR or $LR(1)$ parser, $LALR$ parser uses the $LR(1)$ items of CLR parser & of any shift reduce conflicts are there then it is due to $LR(1)$ parser.
Hence (B) is correct option.

YEAR 2009

Question. 41

Which of the following statements are TRUE ?

I. There exist parsing algorithms for some programming languages

whose complex 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

SOLUTION

I. Statement is true since there are some parsers which take $0(n\log_2 n)$ time for parsing.

II. Completely false, since there is no use of stack which is required for recursion.

III. False

IV. True since both types of optimizations are applied

Hence (B) is correct option.

YEAR 2010

Question. 42

What data structure in a complier is used for managing information about variables and their attributes?

(A) Abstract syntax tree (B) Symbol table

(C) Semantic stack (D) Parse table

SOLUTION

Symbol table is used for storing the information about variables and their attributes by compiler.

Hence (B) is correct option.

Question. 43

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

SOLUTION

Dynamic memory allocation is maintained by heap data structure. So to allow dynamic data structure heap is required.
Hence (C) is correct option.

Question. 44

The grammar $S \rightarrow$aSA|bS|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)

SOLUTION

Given grammar

$$S \rightarrow aSA$$
$$S \rightarrow bS$$
$$S \rightarrow c$$

This grammar is not ambiguous so it is $LL(1)$ also $LR(1)$ grammar since all grammars are $LR(1)$.
Hence (C) is correct option.

****

GATE Multiple Choice Questions
For Computer Science

By NODIA and Company

Available in Two Volumes

FEATURES

- The book is categorized into units and the units are sub-divided into chapters.
- Chapter organization for each unit is very constructive and covers the complete syllabus
- Each chapter contains an average of 40 questions
- The questions are standardized to the level of GATE examination
- Solutions are well-explained, tricky and consume less time. Solutions are presented in such a way that it enhances you fundamentals and problem solving skills
- There are a variety of problems on each topic
- Engineering Mathematics is also included in the book