Summary in Graph

Exam Summary (GO Classes CS Test Series 2025 | Compiler Design | Subject Wise Test)

Qs. Attempted:	0 0 + 0	Correct Marks:	0 0 + 0
Correct Attempts:	0 0 + 0	Penalty Marks:	0 0 + 0
Incorrect Attempts:	0 0 + 0	Resultant Marks:	0 0 + 0

Total Questions:	25 15 + 10
Total Marks:	35 15 + 20
Exam Duration:	90 Minutes
Time Taken:	0 Minutes

EXAM RESPONSE EXAM STATS FEEDBACK

Technical

Q #1 Multiple Choice Type Award: 1 Penalty: 0.33 Compiler Design

Consider the grammar: $S \rightarrow 0\ \mathrm{S}\ 1 \mid 01$. Given the input string 000111, which of the following sequences of actions correctly represents the bottom-up parse using shift and reduce operations?

A. Shift, Shift, Shift, Shift, Reduce $S \rightarrow 01$, Shift, Reduce $S \rightarrow 0S1$, Shift, Reduce $S \rightarrow 0S1$, Accept
B. Shift, Shift, Shift, Reduce $S \rightarrow 0S1$, Shift, Reduce $S \rightarrow 01$, Reduce $S \rightarrow 0S1$, Accept
C. Shift, Shift, Shift, Shift, Reduce $S \rightarrow 01$, Reduce $S \rightarrow 0S1$, Reduce $S \rightarrow 0S1$, Accept
D. Shift, Shift, Reduce $S \rightarrow 01$, Shift, Reduce $S \rightarrow 0S1$, Shift, Reduce $S \rightarrow 0S1$, Accept

Your Answer: Correct Answer: A Not Attempted Time taken: 00min 00sec Discuss

Q #2 Multiple Choice Type Award: 1 Penalty: 0.33 Compiler Design

Match each language rule with the appropriate compiler phase that should verify it:

Laguage Rule	Compiler Phase
1. A function is called with the correct number of arguments	A. Lexical (scanner)

Laguage Rule	**Compiler Phase**
2. Underscore characters (_) may appear in the middle of identifiers but not at the beginning or end	B. Semantic
3. Assignment statements must end with a semicolon (;)	C. Syntactic (parser)

A. 1-A, 2-B, 3-C
B. 1-B, 2-A, 3-C
C. 1-C, 2-A, 3-B
D. 1-A, 2-C, 3-B

Your Answer: **Correct Answer: B** Not Attempted Time taken: 00min 00sec Discuss

Q #3 Multiple Select Type Award: 1 Penalty: 0 Compiler Design

Consider the grammar G defined as:

$S \to$ 0S1 | 01

Which of the following is/are possible viable prefixes for grammar G ?

A. 01
B. 001
C. 0S
D. 00S11

Your Answer: **Correct Answer: A;B;C** Not Attempted Time taken: 00min 00sec Discuss

Q #4 Multiple Select Type Award: 1 Penalty: 0 Compiler Design

In the context of a shift-reduce parser, given the stack content "AcBd," where A and B are non-terminals and c and d are terminals, which of the following could represent possible viable prefixes for the parser?

A. Ac
B. Bd
C. AcB
D. d

Your Answer: **Correct Answer: A;C** Not Attempted Time taken: 00min 00sec Discuss

Q #5 Multiple Choice Type Award: 1 Penalty: 0.33 Compiler Design

Consider the following $\mathrm{LR}(0)$ items of some unknown grammar. What can we infer about these LR(0) items?

1. $A \to Bc\cdot$

2. $B \to de\cdot$

3. $C \to f \cdot B$

A. Item 1 and Item 2 could be part of the same $\mathrm{LR}(0)$ automaton state.
B. Item 1 and Item 3 could be part of the same $\mathrm{LR}(0)$ automaton state.

C. Item 2 and Item 3 could never be part of the same $\mathrm{LR}(0)$ automaton state.
D. Items 1, 2, and 3 must belong to different $\mathrm{LR}(0)$ automaton states.

Your Answer: **Correct Answer: D** Not Attempted Time taken: 00min 00sec Discuss

Q #6 Multiple Select Type Award: 1 Penalty: 0 Compiler Design

Consider the $\mathrm{LR}(0)$ automaton for some grammar, and assume the automaton is in the following state containing the items:

1. $S \rightarrow Sa \cdot b$
2. $B \rightarrow a \cdot$

What must be TRUE about the stack of the $\mathrm{LR}(0)$ parser?

A. The top of the stack must be 'a'.
B. The top of the stack must be 'Sa' (a is on top).
C. The top of the stack must contain 'b'.
D. This state has Shift-Reduce conflict.

Your Answer: **Correct Answer: A;B;D** Not Attempted Time taken: 00min 00sec Discuss

Q #7 Multiple Select Type Award: 1 Penalty: 0 Compiler Design

Consider the following grammar in which S is the start non-terminal and in which the productions for the non-terminals A, B and C are not shown. There are no other productions for S.

$$\begin{aligned}
&\mathrm{S} \rightarrow \mathrm{A\ B} \mid \mathrm{A\ C} \\
&\mathrm{A} \rightarrow \cdots \\
&\mathrm{B} \rightarrow \cdots \\
&\mathrm{C} \rightarrow \cdots
\end{aligned}$$

Which of the following option(s) is/are CORRECT about the entries in row corresponding to S in $\mathrm{LL}(1)$ parsing table ?

A. If $\mathrm{First}(\mathrm{A}) \neq \{\varepsilon\}$ then row corresponding to S always have multiple entries
B. If $\mathrm{First}(\mathrm{A}) = \{\varepsilon\}, \varepsilon \notin \mathrm{First}(B)$ and $\varepsilon \notin \mathrm{First}(\mathrm{C})$ then row corresponding to S will have single entry if and only if $\mathrm{First}(\mathrm{B}) \cap \mathrm{First}(\mathrm{C}) = \varnothing$
C. If $\mathrm{First}(\mathrm{A}) = \{\varepsilon\}, \varepsilon \in \mathrm{First}(\mathrm{B})$ and $\varepsilon \in \mathrm{First}(\mathrm{C})$ then row corresponding to S always have multiple entries
D. If $\varepsilon \in \mathrm{First}(\mathrm{A})$ then row corresponding to S always have multiple entries

Your Answer: **Correct Answer: A;B;C** Not Attempted Time taken: 00min 00sec Discuss

Q #8 Multiple Select Type Award: 1 Penalty: 0 Compiler Design

Consider the grammar $\mathrm{S} \rightarrow \mathrm{Sa} \mid \mathrm{b}$

Which of the following is/are viable prefixes?

A. S
B. Sa
C. b
D. a

Your Answer: | Correct Answer: A;B;C | Not Attempted | Time taken: 00min 00sec | Discuss

Q #9 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | Compiler Design

Expressions in a certain language can be described by a grammar as follows:

$$\begin{gathered} < \text{ expression } >::=< \text{ term } >|< \text{ expression } > op_1 < \text{ term } > \\ < \text{ term } >::=< \text{ item } >|< \text{ term } > op_2 < \text{ item } \rangle \\ < \text{ item } >:::=< \text{ variable } >|< \text{ number } \rangle \end{gathered}$$

This syntax is most appropriate when the order of evaluation is

A. from left to right always
B. from left to right, but op_1 takes precedence over op_2
C. from left to right, but op_2 takes precedence over op_1
D. in any order, but op_1 takes precedence over op_2

Your Answer: | Correct Answer: C | Not Attempted | Time taken: 00min 00sec | Discuss

Q #10 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | Compiler Design

Consider the following operations on linear sequences of characters:

$head(x) =$ first character of x unless $x = \Lambda$, in which case $head(x)$ is not defined.

$\text{tail}(x) =$ all of x except the first character. ($= \Lambda$ if x has < 2 characters.)

$join(x, y) =$ sequence that is the concatenation of x and y.
For example, if $x = 'abc'$, then

head $(x) = 'a'$, tail $(x) = 'bc'$, $tail(head(x)) = \Lambda$, and $join\,(x,' \quad de') = 'abcde'$

Λ is the null string.

Which of the following reverses the order of characters in a string of two or more characters if reverse $(x) = x$, where x is a string with a single character?

A. reverse $(x) = \text{join(reverse (tail}(x)\text{), head }(x))$
B. reverse $(x) = \text{join}(\text{tail}(\text{reverse}(x)), \text{head}(x))$
C. $reverse(x) = join(tail(x), head(x))$
D. reverse $(x) = \text{reverse(join (head}(x)\text{, tail}(x)))$

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #11 | Multiple Select Type | Award: 1 | Penalty: 0 | Compiler Design

A context-free grammar G is ambiguous iff

A. G is not in Chomsky normal form.
B. some string $w \in \mathrm{L}(\mathrm{G})$ has at least two different derivations.
C. some string $w \in \mathrm{L}(\mathrm{G})$ has at least two different parse trees.
D. every string $w \in \mathrm{L}$ (G) has at least two different parse trees.

Your Answer: | Correct Answer: C | Not Attempted | Time taken: 00min 00sec | Discuss

Q #12 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | Compiler Design

In $\mathrm{SLR}(1)$ parser,

In state I we have the items $\mathrm{A} \rightarrow \alpha.$ and $\mathrm{B} \rightarrow \delta.$, $\mathrm{First}(\mathrm{A})$, $\mathrm{Follow}(\mathrm{A})$ and $\mathrm{Follow}(\mathrm{B})$ contains the symbol 'a'. This leads to

A. Shift-reduce conflict
B. Reduce-reduce conflict
C. Both shift-reduce and reduce-reduce conflicts
D. No conflicts

Your Answer: | Correct Answer: B | Not Attempted | Time taken: 00min 00sec | Discuss

Q #13 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | Compiler Design

Which of the following three-Address code satisfies the below given expression?

$$a + (b \times a) + b + c + d$$

A. $t_1 = a + b, t_2 = b \times t_1, t_3 = t_2 + c, t_4 = t_3 + d$
B. $t_1 = a + b, t_2 = b \times t_1, t_3 = a + t_2, t_4 = t_3 + c, t_5 = t_4 + d$
C. $t_1 = a \times b, t_2 = b \times t_1, t_3 = a + t_2, t_4 = t_3 + c, t_5 = t_4 + d$
D. None of these

Your Answer: | Correct Answer: D | Not Attempted | Time taken: 00min 00sec | Discuss

Q #14 | Multiple Choice Type | Award: 1 | Penalty: 0.33 | Compiler Design

Consider the following grammar.

$$\begin{aligned} \mathbf{X} &::= \ \mathbf{a\,Y} \mid \mathbf{Z} \\ \mathbf{Y} &::= \ \mathbf{a} \mid \mathbf{c} \\ \mathbf{Z} &::= \ \mathbf{b\,Y} \end{aligned}$$

Which of the following is true for this grammar?

A. it is $\mathrm{LL1}$ but Not $\mathrm{LR1}$.
B. It is $\mathrm{LR1}$ but Not $\mathrm{LL1}$.
C. It is both $\mathrm{LL1}$ and $\mathrm{LR1}$.
D. It is neither $\mathrm{LL1}$, nor $\mathrm{LR1}$.

Your Answer: | Correct Answer: C | Not Attempted | Time taken: 00min 00sec | Discuss

Q #15 | Multiple Select Type | Award: 1 | Penalty: 0 | Compiler Design

Which of the following statement(s) is/are FALSE?

A. If a given grammar is $\mathrm{LR}(1)$ it is also $\mathrm{LR}(0)$.
B. If a given grammar is $\mathrm{LL}(1)$ it is also $\mathrm{SLR}(1)$.

C. If a given grammar is $\mathrm{LR}(1)$ it is also $\mathrm{LALR}(1)$.
D. It is possible for a grammar to be $\mathrm{LL}(1)$ but not $\mathrm{LR}(1)$.

Your Answer: | Correct Answer: A;B;C;D | Not Attempted | Time taken: 00min 00sec | Discuss

Q #16 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Compiler Design

Consider the grammar with associated semantic actions. Each attribute of a non-terminal is either inherited or synthesized.

$\mathrm{G} \rightarrow \mathrm{F}$	$\mathrm{G.p} = \mathrm{F.p}$
$\mathrm{F} \rightarrow \mathrm{F}_1 \wedge \mathrm{F}_2$	$\mathrm{F.p} = \mathrm{And}(\mathrm{F}_1.\mathrm{p}, \mathrm{F}_2.\mathrm{p})$
$\mathrm{F} \rightarrow \mathrm{F}_1 \wedge \mathrm{F}_2$	$\mathrm{F.p} = \mathrm{Or}(\mathrm{F}_1.\mathrm{p}, \mathrm{F}_2.\mathrm{p})$
$\mathrm{F} \rightarrow \neg \mathrm{F}_1$	$\mathrm{F.p} = \mathrm{Neg}(\mathrm{F}_1.\mathrm{p})$
$\mathrm{F} \rightarrow \mathrm{F}_1 \implies \mathrm{F}_2$	$\mathrm{F.p} = \mathrm{Or}(\mathrm{Not}(\mathrm{F}_1.\mathrm{p}), \mathrm{F}_2.\mathrm{p})$
$\mathrm{F} \rightarrow (\mathrm{F}_1)$	$\mathrm{F.p} = \mathrm{F}_1.\mathrm{p}$
$\mathrm{F} \rightarrow \mathrm{id}$	$\mathrm{F.p} = \mathrm{id.lexeme}$

Determine whether each attribute in the grammar is inherited or synthesized.

A. Both G.p and F.p are synthesized.
B. Both G.p and F.p are inherited.
C. G.p is inherited and $F.p$ is synthesized.
D. G. P is synthesized and $F.\mathrm{p}$ is inherited.

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 00sec | Discuss

Q #17 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Compiler Design

Consider the following grammar and its semantic rules:

1. $\mathrm{A} \rightarrow \mathrm{LM}$

- L.i := f1(A.i)
- M.i := f2(L.s)
- A.s := f3(M.s)

2. $\mathrm{A} \rightarrow \mathrm{QR}$

- R.i := f4(A.i)
- Q.i := f5(R.s)
- A.i := f6(Q.s)

Based on the given semantic rules, determine the nature of the syntax-directed definition for the grammar:

A. S-attributed but not L-attributed
B. L-attributed but not S -attributed
C. Both S-attributed and L-attributed
D. None of the above

Your Answer: | Correct Answer: D | Not Attempted | Time taken: 00min 00sec | Discuss

Q #18 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Compiler Design

Consider the $\mathrm{LL}(1)$ attribute grammar given below:

$$\langle \mathrm{E} \rangle \quad \rightarrow \quad \langle \mathrm{T} \rangle \quad \langle \mathrm{TT} \rangle \qquad \mathrm{TT.st} := \mathrm{T.\,val}; \text{ E.val} := \text{TT.val}$$

$$\langle \mathrm{T} \rangle \quad \rightarrow \quad \mathrm{unsigned\_int} \qquad \mathrm{T.val} = \mathrm{unsigned\_int.val}$$

$$\langle \mathrm{TT}_1 \rangle \rightarrow + \quad \langle \mathrm{T} \rangle \quad \langle \mathrm{TT}_2 \rangle \qquad \mathrm{TT}_2.\,\mathrm{st} := \mathrm{TT}_1.\,\mathrm{st} + \mathrm{T.val}; \mathrm{TT}_1.\,\mathrm{val} := \mathrm{TT}_2.\,\mathrm{val}$$

$$\langle \mathrm{TT} \rangle \rightarrow \quad \epsilon \qquad \mathrm{TT.val} := \mathrm{TT.st}$$

Which of the following attributes are synthesized or inherited?

A. E.val is synthesized; T.val is inherited.
B. T.val is synthesized; TT.val is inherited.
C. Both E.val and T.val are synthesized; TT.st is inherited.
D. TT.val is synthesized; T.val is inherited.

Your Answer: | **Correct Answer: C** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #19 | Multiple Select Type | Award: 2 | Penalty: 0 | Compiler Design

The grammar $S \rightarrow aSa|aa$ generates all even-length strings of $a's$. We can devise a recursive-descent parser with backtrack for this grammar. If we choose to expand using the production $S \rightarrow aa$ first, we will only recognize the string "aa". Therefore, any reasonable recursive-descent parser will try $S \rightarrow aSa$ first.

Which of the following strings will be recognized by this parser?

A. aa (2 a's)
B. aaaa (4 a's)
C. aaaaaa (6 a's)
D. aaaaaaaa (8 a's)

Your Answer: | **Correct Answer: A;B;D** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #20 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Compiler Design

Consider the following grammar:
$S \rightarrow Aa|bAc|Bc \mid bBa$
$A \rightarrow d$
$B \rightarrow d$

Which of the following statements correctly describes about the properties of this grammar?

A. The grammar is both $\mathrm{LR}(1)$ and $\mathrm{LALR}(1)$.
B. The grammar is $\mathrm{LR}(1)$ but not $\mathrm{LALR}(1)$.
C. The grammar is neither $\mathrm{LR}(1)$ nor $\mathrm{LALR}(1)$.
D. The grammar is $\mathrm{SLR}(1)$ grammar.

Your Answer: | **Correct Answer: B** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #21 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Compiler Design

Consider the following grammar, where $\{a, b, c\}$ are terminal symbols:

$$\begin{aligned} S &\rightarrow aXab \mid Y \\ X &\rightarrow bYa \mid \varepsilon \\ Y &\rightarrow Sc \end{aligned}$$

Suppose an $\mathrm{LR}(1)$ parser starts parsing the string: abaabccaab.

What would be the content of the stack immediately before shifting the first ' c ' from the left?

A. $abaXab$
B. abS
C. $abSc$
D. abY

Your Answer: | Correct Answer: B | Not Attempted | Time taken: 00min 00sec | Discuss

Q #22 | Numerical Type | Award: 2 | Penalty: 0 | Compiler Design

```
 1. X = 2;
 2. Y = 10;
 3. Y = X * Y;
 4. A = Y * X - 2 * Y;
 5. B = X / 2 + Y;
 6. Z = 10;
 7. if (B < Z) goto 12
 8. D = Y - Z * 3;
 9. Q = Y - 8;
10. Z = Z - Q;
 11. goto 7;
 12. X = X + Y;
 13. if (X < Z*100) goto 4;
 14. Y = D;
 15. Halt;
```

Identify the basic blocks and draw the CFG(control flow graph) for the code above. Let the number of basics blocks(nodes) be M and edges be N then $\mathrm{M}+\mathrm{N}$ is? (Do Not count/include the special entry and exit nodes in the CFG)

Your Answer: | Correct Answer: 13 | Not Attempted | Time taken: 00min 00sec | Discuss

Q #23 | Multiple Choice Type | Award: 2 | Penalty: 0.67 | Compiler Design

If the expression $((2+3)*4+5*(6+7)*8)+9$ is evaluated with $*$ having precedence over $+$, then the value obtained is the same as the value of which of the following prefix expressions?

A. $++*+234**5+6789$
B. $+*++234**5+6789$
C. $*++234**5++6789$
D. $*+++234**5+6789$

Your Answer: | Correct Answer: A | Not Attempted | Time taken: 00min 03sec | Discuss

Q #24 | Multiple Select Type | Award: 2 | Penalty: 0 | Compiler Design

Our lexer algorithm does the following: When using regular expressions to scan an input, we resolve conflicts by taking the largest possible match at any point.

Consider the following two scanners, for each scanner, we have given tokens and their associated regular expressions (Regular expression followed by Token name in the curly braces).

Scanner 1:

```
    a { return A ; }
    aba { return B ; }
    bab { return C ; }
```

Scanner 2:

```
    b { return A ; }
    aba { return B ; }
    bab { return C ; }
```

For which of the (scanner, string) pair, our usual lexer algorithm, taking the largest match at every step, will fail to break the string in a way in which each piece matches one of the regular expressions.

A. (Scanner 1, abab)
B. (Scanner 1, baba)
C. (Scanner 2, abab)
D. (Scanner 2, baba)

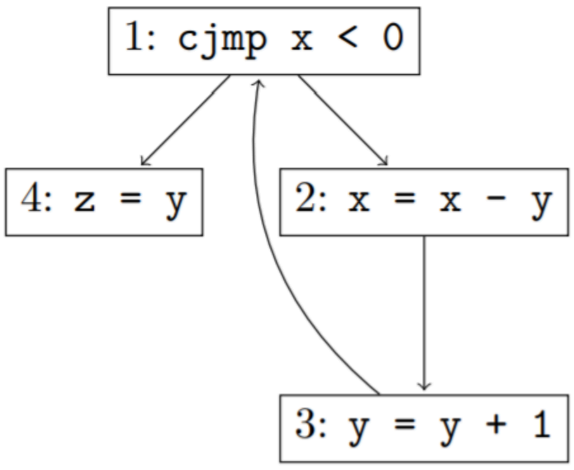
Your Answer: | **Correct Answer: A;D** | Not Attempted | Time taken: 00min 00sec | Discuss

Q #25 | Multiple Select Type | Award: 2 | Penalty: 0 | Compiler Design

For a statement S in a program, in the context of liveness analysis, the following sets are defined:

- $\mathrm{USE}(\mathrm{S})$: the set of variables used in S
- $\mathrm{IN}(\mathrm{S})$: the set of variables that are live at the entry of S
- $\mathrm{OUT}(\mathrm{S})$: the set of variables that are live at the exit of S

Consider the following control flow graph of a program :



Which of the following is true?

A. $\text{out}[4] = \{\}$
B. $\text{in}[4] = \{\text{y}\}$
C. $\text{in}[1] = \{\text{x}, \text{y}\}$
D. $\text{out}[2] = \{\text{y}\}$

Your Answer: | **Correct Answer: A;B;C** | Not Attempted | Time taken: 00min 00sec | Discuss