

((MARKS)) (1/2/3...)	1
((QUESTION))	In full binary search tree every internal node has exactly two children. If there are 100 leaf nodes in the tree, how many internal nodes are there in the tree?
((OPTION_A))	25
((OPTION_B))	49
((OPTION_C))	99
((OPTION_D))	100
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	If a node having two children is to be deleted from binary search tree, it is replaced by its
((OPTION_A))	In-order predecessor
((OPTION_B))	In-order successor
((OPTION_C))	Pre-order predecessor
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary search tree is formed from the sequence 6, 9, 1, 2, 7, 14, 12, 3, 8, 18. The minimum number of nodes required to be added in to this tree to form an extended binary tree is?
((OPTION_A))	3
((OPTION_B))	6
((OPTION_C))	8
((OPTION_D))	11
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Run time for traversing all the nodes of a binary search tree with n nodes and printing them in an order is
((OPTION_A))	$O(n \lg(n))$
((OPTION_B))	$O(n)$
((OPTION_C))	$O(\sqrt{n})$
((OPTION_D))	$O(\log(n))$
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In Binary trees nodes with no successor are called
((OPTION_A))	End nodes
((OPTION_B))	Terminal nodes
((OPTION_C))	Final nodes
((OPTION_D))	Last nodes
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	If node N is a terminal node in a binary tree then its
((OPTION_A))	Right tree is empty
((OPTION_B))	Left tree is empty
((OPTION_C))	Both left & right sub trees are empty
((OPTION_D))	Root node is empty
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Which of the following ways below is a pre order traversal?
((OPTION_A))	Root->left sub tree->right sub tree
((OPTION_B))	Root-> right sub tree ->left sub tree
((OPTION_C))	right sub tree->left sub tree-> Root
((OPTION_D))	left sub tree->Root sub tree->Root
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Which of the following ways below is a pre order traversal?
((OPTION_A))	Root->left sub tree->right sub tree
((OPTION_B))	Root-> right sub tree ->left sub tree
((OPTION_C))	right sub tree->left sub tree-> Root
((OPTION_D))	left sub tree ->Root->Right sub tree
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Which of the following statements hold true for binary trees?
((OPTION_A))	The left subtree of a node contains only nodes with keys less than the node's key
((OPTION_B))	The right subtree of a node contains only one nodes with key greater than the node's key.
((OPTION_C))	Both a and b above
((OPTION_D))	Noth left and right subtree nodes contains only nodes with keys less than the node's key
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Which of the following ways below is a postorder traversal?
((OPTION_A))	Root->left sub tree->right sub tree
((OPTION_B))	Root-> right sub tree ->left sub tree
((OPTION_C))	left sub tree-> right sub tree->Root
((OPTION_D))	left sub tree ->Root->Right sub tree
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree can be converted in to its mirror image by traversing it in
((OPTION_A))	Inorder
((OPTION_B))	Preorder
((OPTION_C))	Postorder
((OPTION_D))	Anyorder
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The in order traversal of tree will yield a sorted listing of elements of tree in
((OPTION_A))	Binary trees
((OPTION_B))	Binary search trees
((OPTION_C))	Heaps
((OPTION_D))	None of above
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION)) left	<pre> 14 / \ 2 16 /\ 1 5 / 4 </pre> <p>Suppose we remove the root, replacing it with something from the tree. What will be the new root?</p>
((OPTION_A))	1
((OPTION_B))	2
((OPTION_C))	4
((OPTION_D))	16
((CORRECT_C HOICE)) (A/B/ C/D)	D
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The inorder traversal of following tree will give a sorted listing of elements in the tree_____.
((OPTION_A))	Binary tree
((OPTION_B))	Binary search tree
((OPTION_C))	AVL tree
((OPTION_D))	Expression tree
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The time complexity of searching an element from a binary search tree is_____.
((OPTION_A))	O(n)
((OPTION_B))	O(n ²)
((OPTION_C))	O(n log n)
((OPTION_D))	O(log n)
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	One can make exact replica of binary search tree by traversing it in_____.
((OPTION_A))	inorder
((OPTION_B))	preorder
((OPTION_C))	Postorder
((OPTION_D))	Any order
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Preorder traversal is nothing but_____.
((OPTION_A))	Depth first search
((OPTION_B))	Breadth first search
((OPTION_C))	Linear order
((OPTION_D))	Topological order
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Maximum number of nodes at each level i are_____.
((OPTION_A))	2
((OPTION_B))	3
((OPTION_C))	$I+1$
((OPTION_D))	$2i+1$
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Let T be a binary search tree with 14 nodes and 60 as external path length. Then The internal path length is_____.
((OPTION_A))	50
((OPTION_B))	11
((OPTION_C))	32
((OPTION_D))	28
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Let T Be binary search tree with 10 nodes and 30 as internal path length. Then the external path length is_____.
((OPTION_A))	50
((OPTION_B))	11
((OPTION_C))	32
((OPTION_D))	28
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Consider the following tree.
((OPTION_A))	10
((OPTION_B))	11
((OPTION_C))	12
((OPTION_D))	14
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The average number of comparisons required for a successful search in a binary search tree with n nodes is_____.
((OPTION_A))	$(I+n/n)$
((OPTION_B))	$(I+2n)$
((OPTION_C))	I/n
((OPTION_D))	$E/(I+n)^2$
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The average number of comparisons required for unsuccessful search in a binary search tree with n nodes is_____.
((OPTION_A))	$(E+n/n)$
((OPTION_B))	$(E+2n)$
((OPTION_C))	E/n
((OPTION_D))	$E/(n+1)$
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Construct a BST for the elements 5,3,7,2,4,8. The average number of comparisons required for a successful search in binary search tree with these nodes is_____.
((OPTION_A))	20
((OPTION_B))	1.33
((OPTION_C))	2.33
((OPTION_D))	32.66
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Which of the following is correct? (I)If each node contains a parent field then it is not necessary to use either Stack Or thread. (II)Traversal using parent pointer is more efficient than the traversal threaded tree. (III)A in-threaded binary tree is defined as binary tree that is left in threaded and right in threaded.
((OPTION_A))	I and II
((OPTION_B))	I and III
((OPTION_C))	II and III
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary search tree is created by inserting following integers -50 ,14 ,65 ,5,20,57,91,3,8,37,60,25 The number of nodes in left and right subtree are_____.
((OPTION_A))	(7,4)
((OPTION_B))	(4,7)
((OPTION_C))	(3,6)
((OPTION_D))	(6,3)
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION))	

((OPTIONAL))	
--------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	If the post order traversal gives the expression $ab+gde-/*$ then label of the Nodes 1,2,3..... will be_____.
((OPTION_A))	$*,+/,a,,b,g,-,d,e$
((OPTION_B))	$A,b,g,d,e,*,+/,.-$
((OPTION_C))	$A,+,b,*,g,/,d,-,e$
((OPTION_D))	$*,+,a,b,/,g,-,d,e$
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The number of null branches in a binary tree with 20 nodes is_____.
((OPTION_A))	18
((OPTION_B))	19
((OPTION_C))	20
((OPTION_D))	21
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Suppose the numbers 7,5,1,8,3,6,0,9,4,2 are inserted in that order into an Initially empty binary search tree. The binary search tree uses the usual Ordering On natural numbers. What is the in-order traversal sequence of the Tree?
((OPTION_A))	0 2 4 3 1 6 5 9 8 7
((OPTION_B))	0 1 2 3 4 5 6 7 8 9
((OPTION_C))	7 5 1 0 3 2 4 6 8 9
((OPTION_D))	9 8 6 4 2 3 0 1 5 7
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	What would be the depth of a tree whose level is 7 ?
((OPTION_A))	8
((OPTION_B))	4
((OPTION_C))	9
((OPTION_D))	7
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	How many nodes a complete binary tree of level 5 have ?
((OPTION_A))	15
((OPTION_B))	16
((OPTION_C))	31
((OPTION_D))	32
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	If a node in binary search tree has two children, then its inorder Predecessor has_____.
((OPTION_A))	No left child
((OPTION_B))	No right child
((OPTION_C))	Two children
((OPTION_D))	No child
((CORRECT_C HOICE)) (A/B/ C/D)	A
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	If a node in binary search tree has two children, then its inorder successor has_____.
((OPTION_A))	No left child
((OPTION_B))	No right child
((OPTION_C))	Two children
((OPTION_D))	No child
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In order to order to get the information stored in a binary search tree in The descending order, one should recursively traverse it in the Following order_____.
((OPTION_A))	root , right sub tree ,left sub tree
((OPTION_B))	right sub tree , root left sub tree
((OPTION_C))	right sub tree , left sub tree, root
((OPTION_D))	root , left sub tree , right sub tree
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Breadth first search_____.
((OPTION_A))	Scans each incident node along with the child nodes
((OPTION_B))	Scans all the nodes in preorder manner
((OPTION_C))	Scans all the nodes in random manner
((OPTION_D))	Scans all the incident edges before moving other nodes
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>Consider the following code.</p> <pre> Int count(bintree *temp, int cnt) { Static int i = 0; If(temp != NULL) { If ((temp->left == NULL) && (temp->right == NULL)) { Printf("%d", temp->data); i ++; return i; } } Else { Count(temp->left,i); Count(temp->right,i); } } </pre> <p>It returns ---</p>
((OPTION_A))	Total number of nodes
((OPTION_B))	Total number of leaf nodes
((OPTION_C))	Total number of internal nodes
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>A binary search tree stores value in the range 50 to 550. Consider the owing</p> <ol style="list-style-type: none"> 1. 78 540 100 440 290 280 310 2. 50 99 120 198 245 381 480 3. 145 250 510 390 350 270 307 4. 550 150 507 398 463 402 270 <p>Which of the following statement is true</p>
((OPTION_A))	1,2 and 4 are inorder sequence of three different BST
((OPTION_B))	1 is preorder sequence of some BST with 440 as root
((OPTION_C))	2 is an inorder sequence of some BST where 120 is the root and 50 is leaf.
((OPTION_D))	4 is postorder sequence of some BST with 150 as root.
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>A binary search is created using the integers 10,8,13,11,12. If we want to represent this thee using the sequential representation then node 11 will be at</p> <p>At location.</p>
((OPTION_A))	6
((OPTION_B))	5
((OPTION_C))	4
((OPTION_D))	3
((CORRECT_C HOICE)) (A/B/ C/D)	A

((EXPLANATI ON)) (OPTIONAL)	
((MARKS)) (1/2/3...)	1
((QUESTION))	In which order the following numbers 12,3,2,4,6,7,5,1 should be inserted in an An empty binary search tree of height 5.
((OPTION_A))	1,2,3,5,12,6,7,4
((OPTION_B))	1,2,3,6,5,7,12,4
((OPTION_C))	1,2,3,4,6,7,12,5
((OPTION_D))	None of these
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In tree creation which is the most suitable data structure?
((OPTION_A))	Arrays
((OPTION_B))	Stack
((OPTION_C))	Queue
((OPTION_D))	Link list
((CORRECT_C HOICE)) (A/B/ C/D)	D
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Construct BST for 10,08,15,12,13,07,09,17,20,18,04,05 what will be the Node of node 17?
((OPTION_A))	9
((OPTION_B))	20
((OPTION_C))	10
((OPTION_D))	15
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The number of nodes in null binary tree are
((OPTION_A))	1
((OPTION_B))	2
((OPTION_C))	0
((OPTION_D))	Null tree is invalid tree
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The preorder sequence of BST is 30,22,17,27,24,38,34,48. What is the Sequence of the same tree?
((OPTION_A))	17,22,24,27,30,34,38,48
((OPTION_B))	30,22,38,17,27,34,48,24
((OPTION_C))	17,22,27,24,34,48,38,30
((OPTION_D))	17,24,27,22,34,48,38,30
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The preorder sequence of BST is 30,22,17,27,24,38,34,48. What is the Order Sequence of the same tree?
((OPTION_A))	17,22,24,27,30,34,38,48
((OPTION_B))	30,22,38,17,27,34,48,24
((OPTION_C))	17,22,27,24,34,48,38,30
((OPTION_D))	17,24,27,22,34,48,38,30
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

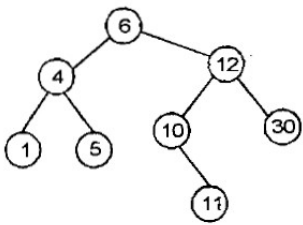
((MARKS)) (1/2/3...)	1
((QUESTION))	The level order traversal of rooted tree can be done by starting from root And Performing.-----
((OPTION_A))	Preorder
((OPTION_B))	Inorder
((OPTION_C))	DFS
((OPTION_D))	BFS
((CORRECT_C HOICE)) (A/B/ C/D)	D
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	When preorder and postorder sequence generates the same output then the tree T must have-----
((OPTION_A))	3 Nodes
((OPTION_B))	2 Nodes
((OPTION_C))	1 Nodes
((OPTION_D))	Any no. Of Nodes
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

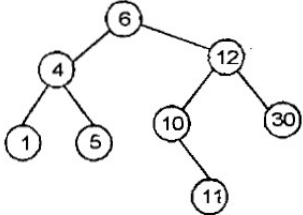
((MARKS)) (1/2/3...)	1
((QUESTION))	In which order the numbers 7,5,1,8,3,2 should be inserted into an empty BST to get inorder and preorder sequence as same.
((OPTION_A))	3,2,1,8,7,5
((OPTION_B))	8,7,5,3,2,1
((OPTION_C))	1,2,3,5,7,8
((OPTION_D))	3,2,8,7,5,1
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The postorder sequence of BST is as given. I,L,K,M,Y,Z,W what will be preorder sequence.
((OPTION_A))	I,K,L,M,W,Y,Z
((OPTION_B))	W,M,K,I,L,Z,Y
((OPTION_C))	W,M,Z,K,I,L,Y
((OPTION_D))	W,M,K,L,I,Z,Y
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The following numbers are inserted into an empty BST in given order 10,2,3,5,15,12,18 what is the height of BST?
((OPTION_A))	2
((OPTION_B))	3
((OPTION_C))	4
((OPTION_D))	6
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>If this tree is used for sorting, then new no 8 should be placed as the</p>  <pre> graph TD 6((6)) --- 4((4)) 6 --- 12((12)) 4 --- 1((1)) 4 --- 5((5)) 12 --- 10((10)) 12 --- 30((30)) 10 --- 11((11)) </pre>
((OPTION_A))	left child of the node labelled 30.
((OPTION_B))	right child of the node labelled 5
((OPTION_C))	right child of the node labelled 30
((OPTION_D))	left child of the node labelled 10
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION))	

((OPTIONAL))	
--------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>If this tree is used for sorting, then new no 35 should be placed as the</p>  <pre> graph TD 6((6)) --- 4((4)) 6 --- 12((12)) 4 --- 1((1)) 4 --- 5((5)) 12 --- 10((10)) 12 --- 30((30)) 10 --- 11((11)) </pre>
((OPTION_A))	left child of the node labelled 30.
((OPTION_B))	right child of the node labelled 5
((OPTION_C))	right child of the node labelled 30
((OPTION_D))	left child of the node labelled 10
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	It is necessary for Huffman encoding tree to be,
((OPTION_A))	AVL tree
((OPTION_B))	Binary tree
((OPTION_C))	Complete binary Tree
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/	B

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In the array representation of binary tree the Right child of root node will be at the locations?
((OPTION_A))	0
((OPTION_B))	1
((OPTION_C))	2
((OPTION_D))	3
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree is a tree in which?
((OPTION_A))	No node can have more than two children
((OPTION_B))	Every node must have two children
((OPTION_C))	A node can have at least two children
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/C/D)	A

((EXPLANATION)) (OPTIONAL)	
-------------------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	binary tree in which if all its levels except possibly the last, have the maximum number of nodes and all the nodes at the last level appear as far left as possible, is known as
((OPTION_A))	Full binary tree
((OPTION_B))	AVL
((OPTION_C))	Threaded binary tree
((OPTION_D))	Complete binary tree
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Consider the binary_tree_node t. Which expression indicates that t represents an empty tree?
((OPTION_A))	(t == NULL)
((OPTION_B))	(t->data() == 0)
((OPTION_C))	(t->data() == NULL)
((OPTION_D))	((t->left() == NULL) && (t->right() == NULL))
((CORRECT_CHOICE)) (A/B/	A

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In a binary tree, certain null entries are replaced by special pointers which point to nodes higher in the tree for efficiency. These special pointers are called
((OPTION_A))	Leaf
((OPTION_B))	branch
((OPTION_C))	Path
((OPTION_D))	Thread
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Select the one true statement.
((OPTION_A))	Every binary tree is either complete or full.
((OPTION_B))	Every complete binary tree is also a full binary tree.
((OPTION_C))	Every full binary tree is also a complete binary tree.
((OPTION_D))	No binary tree is both complete and full.
((CORRECT_CHOICE)) (A/B/C/D)	C

((EXPLANATION)) (OPTIONAL)	
-------------------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	Select the one FALSE statement about binary trees?
((OPTION_A))	Every binary tree has at least one node.
((OPTION_B))	Every non-empty tree has exactly one root node.
((OPTION_C))	Every node has at most two children.
((OPTION_D))	Every non-root node has exactly one parent.
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree of depth d is an almost complete binary tree if?
((OPTION_A))	Each leaf in the tree is either at level d or at level d-1
((OPTION_B))	For any node n in the tree with a right descendant at level d all the left descendants of n that are leaves, are also at level d
((OPTION_C))	Both (A) & (B)
((OPTION_D))	None of the Above
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION))	

(OPTIONAL)	
------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	A full binary tree with $2n+1$ nodes contain?
((OPTION_A))	n leaf nodes
((OPTION_B))	n non-leaf nodes
((OPTION_C))	n-1 leaf nodes
((OPTION_D))	n-1 non-leaf nodes
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The most appropriate matching for the following pairs? X: depth first search 1: heap Y: breadth-first search 2: queue Z: sorting 3: stack
((OPTION_A))	X—1 Y—2 Z-3
((OPTION_B))	X—3 Y—1 Z-2
((OPTION_C))	X—3 Y—2 Z-1
((OPTION_D))	X—2 Y—3 Z-1
((CORRECT_CHOICE)) (A/B/C/D)	C

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The difference between the external path length and the internal path length of a binary tree with n internal nodes is?
((OPTION_A))	1
((OPTION_B))	N
((OPTION_C))	n + 1
((OPTION_D))	2n
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A threaded binary tree is a binary tree in which every node that does not have right child has a thread to its?
((OPTION_A))	Pre-order successor
((OPTION_B))	In-order successor
((OPTION_C))	In-order predecessor
((OPTION_D))	Post-order successor
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION))	

ON)) (OPTIONAL)	
--------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	In binary representation of tree the left child of root node will be at location
((OPTION_A))	0
((OPTION_B))	1
((OPTION_C))	2
((OPTION_D))	3
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In array representation of binary tree right child of root node will be at location
((OPTION_A))	0
((OPTION_B))	1
((OPTION_C))	2
((OPTION_D))	3
((CORRECT_C HOICE)) (A/B/ C/D)	D
((EXPLANATI ON))	

((OPTIONAL))	
--------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	In array representation of binary tree left child of root node will be at location
((OPTION_A))	0
((OPTION_B))	1
((OPTION_C))	2
((OPTION_D))	3
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In array representation of binary tree root node will be at location
((OPTION_A))	0
((OPTION_B))	1
((OPTION_C))	2
((OPTION_D))	3
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION))	

((OPTIONAL))	
--------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	To arrange a tree in ascending order which of the following traversal is
((OPTION_A))	Preorder
((OPTION_B))	Inorder
((OPTION_C))	Postorder
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Depth of complete binary tree with n nodes is
((OPTION_A))	$O(\log n)$
((OPTION_B))	$O(\log n + 1) - 1$
((OPTION_C))	$O(\log n - 1)n - 1$
((OPTION_D))	$O(\log n + 1)$
((CORRECT_CHOICE))	A

HOICE)) (A/B/ C/D)	
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Preorder traversal of binary search tree is as follows: 10,8,7,9,18,11,12. If tree is used for sorting then where will be 15 placed?
((OPTION_A))	Right child of 18
((OPTION_B))	Left child of 9
((OPTION_C))	Left child of 11
((OPTION_D))	Right child of 12
((CORRECT_C HOICE)) (A/B/ C/D)	D
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In Preorder traversal the node which is visted first is
((OPTION_A))	Root
((OPTION_B))	Leftmost
((OPTION_C))	Right most
((OPTION_D))	parent of a right most leaf
((CORRECT_C	A

HOICE)) (A/B/ C/D)	
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Consider a preorder traversal sequence of a tree is : 1 2 4 5 3. In array representation you can store node 4 at location
((OPTION_A))	4
((OPTION_B))	5
((OPTION_C))	6
((OPTION_D))	3
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Number of nodes in strict binary tree with n leaf nodes
((OPTION_A))	N
((OPTION_B))	$2n$
((OPTION_C))	$2n-1$
((OPTION_D))	$2n+1$

((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	For generating a binary search tree from given sequence, following traversal sequences are used:
((OPTION_A))	Inorder
((OPTION_B))	Preorder
((OPTION_C))	Post Order
((OPTION_D))	Preorder and Inorder
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Arrangement in which descendants point to ancestors is
((OPTION_A))	Binary tree
((OPTION_B))	Threaded binary tree
((OPTION_C))	Complete binary tree
((OPTION_D))	Balanced tree
((CORRECT_CHOICE)) (A/B/C/D)	B

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A Game tree represents the
((OPTION_A))	Decision made
((OPTION_B))	Moves made
((OPTION_C))	Data
((OPTION_D))	Comparisons made
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The only node which does not have a predecessor is
((OPTION_A))	Root
((OPTION_B))	Leftmost
((OPTION_C))	Right most
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/	A

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Degree of any node is obtained from
((OPTION_A))	Number of successors
((OPTION_B))	Number of leaf nodes
((OPTION_C))	Number of predecessors
((OPTION_D))	Number of right sub branches
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	While converting the binary tree to general tree the links are added from each node to
((OPTION_A))	Its immediate right child
((OPTION_B))	Its immediate left child
((OPTION_C))	Its leaf node
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/	A

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	If binary search tree traversed in following order recursively: Right, root, left then output sequence will be
((OPTION_A))	Ascending
((OPTION_B))	Descending
((OPTION_C))	Level order
((OPTION_D))	No specific order
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	If binary tree has 50 nodes then number of edges is
((OPTION_A))	51
((OPTION_B))	55
((OPTION_C))	49
((OPTION_D))	50

((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	If a binary tree has 45 internal nodes then binary tree has how Many external nodes?
((OPTION_A))	45
((OPTION_B))	46
((OPTION_C))	50
((OPTION_D))	55
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In complete binary tree represented in array, then for an array element at position i, the right child is at position
((OPTION_A))	$2i$
((OPTION_B))	$2i-1$
((OPTION_C))	$2i+1$

((OPTION_D))	Floor(i/2)
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In complete binary tree represented in array, then for an array element at position i, the left child is at position
((OPTION_A))	$2i$
((OPTION_B))	$2i-1$
((OPTION_C))	$2i+1$
((OPTION_D))	Floor(i/2)
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In complete binary tree represented in array, then for an array element at position i, the root is at position
((OPTION_A))	$2i$
((OPTION_B))	$2i-1$
((OPTION_C))	$2i+1$

((OPTION_D))	Floor($i/2$)
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In binary tree there are 23 external nodes then number of internal nodes are
((OPTION_A))	24
((OPTION_B))	22
((OPTION_C))	21
((OPTION_D))	23
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	For a binary tree if there are n leaf nodes then the number of degree 2 in T is
((OPTION_A))	$\log n$
((OPTION_B))	$n-1$

((OPTION_C))	$\log_2 n$
((OPTION_D))	N
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Following is a property of binary tree
((OPTION_A))	It has $2+n$ links, $n-1$ links to internal nodes and $n+1$ links to external nodes
((OPTION_B))	It has $2-n$ links, $n-1$ links to internal nodes and $n+1$ links to external nodes
((OPTION_C))	It has $2*n$ links, $n-1$ links to internal nodes and $n+1$ links to external nodes
((OPTION_D))	It has $2n$ links, $n+1$ links to internal nodes and $n-1$ links to external nodes
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
-------------------------	---

((QUESTION))	In threaded binary tree NULL pointer is replaced by
((OPTION_A))	Inorder successor or predecessor
((OPTION_B))	Preorder successor or predecessor
((OPTION_C))	Postorder successor or predecessor
((OPTION_D))	Parent
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In expression tree root always contain
((OPTION_A))	Operand
((OPTION_B))	Operator
((OPTION_C))	Opening Paranthesis
((OPTION_D))	Closing Paranthesis
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In expression tree leaf always contain
((OPTION_A))	Operand

((OPTION_B))	Operator
((OPTION_C))	Opening Paranthesis
((OPTION_D))	Closing Paranthesis
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	An expression tree is always
((OPTION_A))	A complete binary tree
((OPTION_B))	Heap
((OPTION_C))	Binary search tree
((OPTION_D))	Expression tree
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
-------------------------	---

((QUESTION))	Heap is implemented as
((OPTION_A))	Threaded binary tree
((OPTION_B))	Complete binary tree
((OPTION_C))	Expression tree
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree has ----- links , --- links to internal nodes and ----- links to
((OPTION_A))	$n-1, 2n, n+1$
((OPTION_B))	$2n, n-1, n+1$
((OPTION_C))	$N+1, 2n, n-1$
((OPTION_D))	$N+1, n-1, 2n$
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

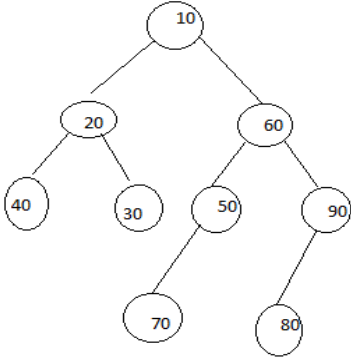
((MARKS))	1
-----------	---

(1/2/3...)	
((QUESTION))	By using which tree recursive method for traversing a tree is avoided?
((OPTION_A))	Binary tree
((OPTION_B))	Balanced tree
((OPTION_C))	Threaded binary tree
((OPTION_D))	Expression tree
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

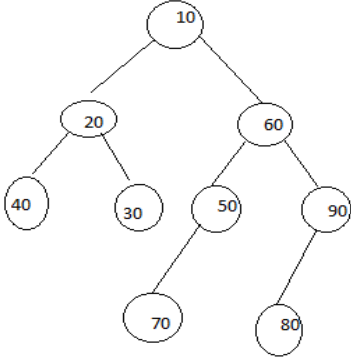
((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree with 14 nodes what is the minimum possible depth of this tree?
((OPTION_A))	1
((OPTION_B))	2
((OPTION_C))	3
((OPTION_D))	4
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree with 14 nodes what is the minimum possible height of this tree?

((OPTION_A))	1
((OPTION_B))	2
((OPTION_C))	3
((OPTION_D))	4
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>What is depth of following tree?</p>  <pre> graph TD 10((10)) --> 20((20)) 10 --> 60((60)) 20 --> 40((40)) 20 --> 30((30)) 60 --> 50((50)) 60 --> 90((90)) 50 --> 70((70)) 90 --> 80((80)) </pre>
((OPTION_A))	2
((OPTION_B))	4
((OPTION_C))	3
((OPTION_D))	1
((CORRECT_CHOICE)) (A/B/C/D)	C

((EXPLANATI ON)) (OPTIONAL)	
-----------------------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>What is height of following tree?</p>  <pre> graph TD 10((10)) --> 20((20)) 10 --> 60((60)) 20 --> 40((40)) 20 --> 30((30)) 60 --> 50((50)) 60 --> 90((90)) 50 --> 70((70)) 90 --> 80((80)) </pre>
((OPTION_A))	2
((OPTION_B))	4
((OPTION_C))	3
((OPTION_D))	1
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Inorder traversal for the tree is

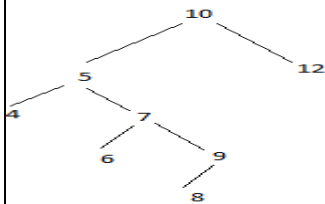
	<pre> graph TD 10((10)) --> 20((20)) 10 --> 60((60)) 20 --> 40((40)) 20 --> 30((30)) 60 --> 50((50)) 60 --> 90((90)) 50 --> 70((70)) 90 --> 80((80)) </pre>
((OPTION_A))	40 20 30 10 70 50 60 80 90
((OPTION_B))	10 20 40 60 50 70 90 80
((OPTION_C))	40 30 20 70 50 80 90 60
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>Preorder traversal for the tree is</p> <pre> graph TD 10((10)) --> 20((20)) 10 --> 60((60)) 20 --> 40((40)) 20 --> 30((30)) 60 --> 50((50)) 60 --> 90((90)) 50 --> 70((70)) 90 --> 80((80)) </pre>
((OPTION_A))	40 20 30 10 70 50 60 80 90
((OPTION_B))	10 20 40 60 50 70 90 80
((OPTION_C))	40 30 20 70 50 80 90 60 10
((OPTION_D))	None

((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Which expression represents that tree t is empty
((OPTION_A))	t->left= NULL and t->right = NULL
((OPTION_B))	t=NULL
((OPTION_C))	t->data=0
((OPTION_D))	t->data =NULL
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	
((MARKS)) (1/2/3...)	1
((QUESTION))	The time required to search a binary search tree varies between
((OPTION_A))	$O(n^2)$ and $O(n)$
((OPTION_B))	$O(n), O(\log n)$
((OPTION_C))	$O(n^2), O(\log n)$
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Number of binary trees with 3 nodes which when traversed in post order fashion gives sequence A B C
((OPTION_A))	6
((OPTION_B))	3
((OPTION_C))	9
((OPTION_D))	5
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	 <p>On deleting 7, it will get replaced by which node?</p>
((OPTION_A))	10
((OPTION_B))	6
((OPTION_C))	8
((OPTION_D))	4
((CORRECT_CHOICE)) (A/B/C/D)	C

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Numbers 1,2,3,.....,n are inserted in a binary tree in some order. In resulting tree, the right subtree of the root contains node. The first number to be inserted in tree must be
((OPTION_A))	P
((OPTION_B))	P+1
((OPTION_C))	n-P
((OPTION_D))	n-P+1
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Numbers 1,2,3,4,5,6,7,8,9 and 10 will be inserted in binary search tree. The right subtree of the root contains 3 nodes. The root node will be
((OPTION_A))	1
((OPTION_B))	4
((OPTION_C))	7
((OPTION_D))	6
((CORRECT_CHOICE)) (A/B/C/D)	C

((EXPLANATION)) (OPTIONAL)	
-------------------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	A complete binary tree of level 5 has how many nodes ?
((OPTION_A))	15
((OPTION_B))	25
((OPTION_C))	63
((OPTION_D))	30
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree having n nodes and depth d will be about complete binary tree if
((OPTION_A))	any node n_d at level less than d-1 has two sons
((OPTION_B))	it contains $\log(d)+1$ nodes
((OPTION_C))	for any node n_d in the tree with a right descendent at level d It must have a left son
((OPTION_D))	all of these
((CORRECT_CHOICE)) (A/B/C/D)	A

((EXPLANATI ON)) (OPTIONAL)	
-----------------------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree in which every non-leaf node has non-empty left and right subtrees is called a strictly binary tree. Such a tree with 10 leaves
((OPTION_A))	Cannot have more than 19 nodes
((OPTION_B))	Has exactly 19 nodes
((OPTION_C))	Has exactly 17 nodes
((OPTION_D))	Cannot have more than 19 nodes
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Number of possible binary trees with 3 nodes is
((OPTION_A))	12
((OPTION_B))	9
((OPTION_C))	14
((OPTION_D))	5
((CORRECT_C HOICE)) (A/B/ C/D)	D
((EXPLANATI	

ON)) (OPTIONAL)	
--------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	The number of possible binary trees with 4 nodes is
((OPTION_A))	12
((OPTION_B))	13
((OPTION_C))	14
((OPTION_D))	15
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	When searching for the key value 60 in a BST, nodes containing key values 10,20,40,50,70,80,90 are traversed not necessarily in the order given. How many different orders are possible in which this key values can occur on the search path from the root node containing the value 60?
((OPTION_A))	35
((OPTION_B))	64
((OPTION_C))	128
((OPTION_D))	5040
((CORRECT_C HOICE)) (A/B/	C

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	<p>The following three are known to be preorder, inorder and postorder sequences of a binary tree. But it is not known which is which</p> <ol style="list-style-type: none"> 1. MBCAFHPYK 2. KAMCBYPFH 3. MABCKYFPH <p>Pick the true statement from the following.</p>
((QUESTION))	1 and 2 are preorder and inorder sequences respectively.
((OPTION_A))	1 and 3 are preorder and postorder sequences respectively.
((OPTION_B))	2 is the inorder sequence.
((OPTION_C))	2 and 3 are the preorder and inorder sequences respectively.
((OPTION_D))	D
((CORRECT_CHOICE)) (A/B/C/D)	
((EXPLANATION)) (OPTIONAL)	
((MARKS)) (1/2/3...)	1
((QUESTION))	In a tree between every pair of vertices there is_____.
((OPTION_A))	Exactly one path
((OPTION_B))	At least two path
((OPTION_C))	N number of paths
((OPTION_D))	Self loop
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION))	

(OPTIONAL)	
------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	Suppose we have numbers between 1 and 100 in a BST and want to search the number 54. Which of the following sequence cannot be the sequence of nodes examined?
((OPTION_A))	10,75,64,43,60,57,54
((OPTION_B))	90,12,68,34,62,45,54
((OPTION_C))	9,85,47,64,43,57,54
((OPTION_D))	79,14,72,56,16,53,54
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Consider the label sequences obtained by the following pairs of traversals on a labeled binary tree. Which of these pairs identify a tree uniquely?
((OPTION_A))	Preorder and postorder
((OPTION_B))	Inorder and postorder
((OPTION_C))	Preorder and levelorder
((OPTION_D))	Levelorder and postorder.
((CORRECT_CHOICE)) (A/B/C/D)	B

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Which one of the following is true about the dummy node in threaded binary tree.
((OPTION_A))	The left pointer of the dummy node points to itself while the right pointer points to the root node.
((OPTION_B))	The left pointer points to the root node and the right pointer points to itself.
((OPTION_C))	The left pointer of the dummy node points to root node of the tree while the right pointer always point to NULL.
((OPTION_D))	The left pointer points to the NULL while the right pointer always points to the dummy node.
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Number of possible binary trees with 5 nodes
((OPTION_A))	10
((OPTION_B))	9
((OPTION_C))	8
((OPTION_D))	32
((CORRECT_CHOICE)) (A/B/C/D)	B

((EXPLANATION)) (OPTIONAL)	
-------------------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree of depth "d" is an almost complete binary tree if
((OPTION_A))	each leaf in the tree is either at level
((OPTION_B))	for any node
((OPTION_C))	Both A and B
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	While converting the general trees to binary tree_____.
((OPTION_A))	The root node of the general tree becomes the root node of the binary tree
((OPTION_B))	The first child of the node is attached as a left child to the current node in the binary tree
((OPTION_C))	The right sibling can be attached as a right child of that node
((OPTION_D))	All of the above
((CORRECT_CHOICE)) (A/B/C/D)	D

((EXPLANATION)) (OPTIONAL)	
-------------------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	In inorder traversal the node which is visited last is
((OPTION_A))	Root
((OPTION_B))	Leftmost
((OPTION_C))	Right most
((OPTION_D))	parent of a right most leaf
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In inorder traversal the node which is visited first is
((OPTION_A))	Root
((OPTION_B))	Leftmost
((OPTION_C))	Right most
((OPTION_D))	parent of a right most leaf
((CORRECT_CHOICE)) (A/B/C/D)	B

((EXPLANATION)) (OPTIONAL)	
-------------------------------	--

((MARKS)) (1/2/3...)	1
((QUESTION))	In inorder traversal the node which is after left subtree is
((OPTION_A))	Root
((OPTION_B))	Leftmost
((OPTION_C))	Right most
((OPTION_D))	parent of a right most leaf
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In a extended-binary tree nodes with 2 children are called
((OPTION_A))	Interior node
((OPTION_B))	Domestic node
((OPTION_C))	Internal node
((OPTION_D))	Inner node
((CORRECT_C	C

HOICE)) (A/B/ C/D)	
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A terminal node in a binary tree is called
((OPTION_A))	Root
((OPTION_B))	Leaf
((OPTION_C))	Child
((OPTION_D))	Branch
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Binary trees with threads are called as
((OPTION_A))	Threaded trees
((OPTION_B))	Pointer trees
((OPTION_C))	Special trees
((OPTION_D))	Special pointer trees
((CORRECT_C HOICE)) (A/B/	A

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	While converting binary tree into extended binary tree, all the original nodes in binary tree are
((OPTION_A))	Internal nodes on extended tree
((OPTION_B))	External nodes on extended tree
((OPTION_C))	Vanished on extended tree
((OPTION_D))	Intermediate nodes on extended tree
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

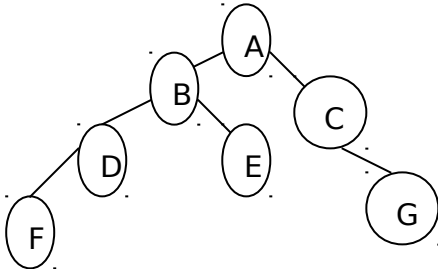
((MARKS)) (1/2/3...)	1
((QUESTION))	The in-order traversal of tree will yield a sorted listing of elements of tree in
((OPTION_A))	binary trees
((OPTION_B))	binary search trees
((OPTION_C))	Heaps
((OPTION_D))	binary heaps
((CORRECT_CHOICE))	B

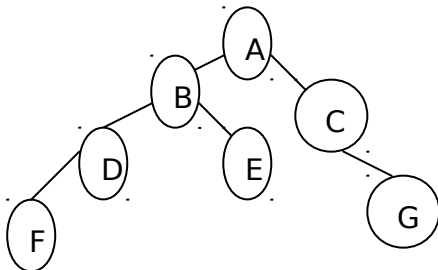
HOICE)) (A/B/ C/D)	
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	If node N is a terminal node in a binary tree then its
((OPTION_A))	Right tree is empty
((OPTION_B))	Left tree is empty
((OPTION_C))	Both left & right sub trees are empty
((OPTION_D))	Root node is empty
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

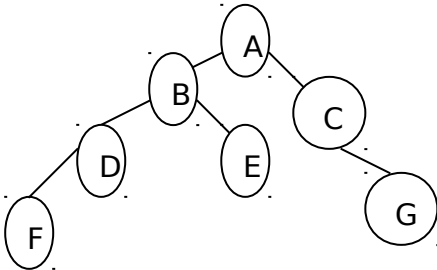
((MARKS)) (1/2/3...)	1
((QUESTION))	Which out of these is a non-linear data-structure:
((OPTION_A))	arrays
((OPTION_B))	linked-lists
((OPTION_C))	queues
((OPTION_D))	tree
((CORRECT_C HOICE)) (A/B/	D

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>The height of the following binary tree is:</p>  <pre> graph TD A((A)) --- B((B)) A --- C((C)) B --- D((D)) B --- E((E)) D --- F((F)) C --- G((G)) </pre>
((OPTION_A))	3
((OPTION_B))	4
((OPTION_C))	5
((OPTION_D))	2
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>The height of the left subtree of following binary tree is:</p>  <pre> graph TD A((A)) --- B((B)) A --- C((C)) B --- D((D)) B --- E((E)) D --- F((F)) C --- G((G)) </pre>

((OPTION_A))	3
((OPTION_B))	4
((OPTION_C))	5
((OPTION_D))	2
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>The height of the right subtree of following binary tree is:</p>  <pre> graph TD A((A)) --- B((B)) A --- C((C)) B --- D((D)) B --- E((E)) D --- F((F)) C --- G((G)) </pre>
((OPTION_A))	3
((OPTION_B))	4
((OPTION_C))	5
((OPTION_D))	2
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	A binary search tree in which the nodes have been inserted in the following order:60,55,95,40,30,100,35, the node with the value 47 will be inserted to
-------------------------	---

	the:
((QUESTION))	right of node with value 40
((OPTION_A))	right of node with value 55
((OPTION_B))	right of node with value 35
((OPTION_C))	left of node with value 30
((OPTION_D))	A
((CORRECT_CHOICE)) (A/B/C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A tree having any number of nodes:
((OPTION_A))	binary tree
((OPTION_B))	general tree
((OPTION_C))	B- tree
((OPTION_D))	AVL tree
((CORRECT_CHOICE)) (A/B/	B

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A binary tree grows at the
((OPTION_A))	root
((OPTION_B))	leaves
((OPTION_C))	braches
((OPTION_D))	any of the above
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The leaf nodes of a tree have height equal to:
((OPTION_A))	height of the tree
((OPTION_B))	zero
((OPTION_C))	one
((OPTION_D))	none of these
((CORRECT_CHOICE))	B

HOICE)) (A/B/ C/D)	
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The root nodes of a tree have depth equal to:
((OPTION_A))	height of the tree
((OPTION_B))	zero
((OPTION_C))	one
((OPTION_D))	none of these
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	All leaf nodes of a tree are termed as:
((OPTION_A))	terminal nodes
((OPTION_B))	non-terminal nodes

((OPTION_C))	child nodes
((OPTION_D))	internal nodes
((CORRECT_C HOICE)) (A/B/ C/D)	A
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The root node is
((OPTION_A))	terminal node
((OPTION_B))	internal nodes
((OPTION_C))	child node
((OPTION_D))	none of the above
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

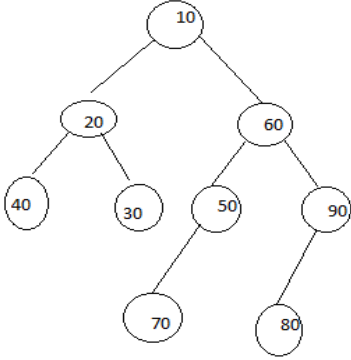
((MARKS)) (1/2/3...)	1
((QUESTION))	In the following post-order traversal of a binary tree: E,C,K,A,H,B,G,D,F, the root node is:
((OPTION_A))	E
((OPTION_B))	H
((OPTION_C))	F

((OPTION_D))	D
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The root nodes of a tree have depth equal to:
((OPTION_A))	depth of the tree
((OPTION_B))	zero
((OPTION_C))	one
((OPTION_D))	none of these
((CORRECT_C HOICE)) (A/B/ C/D)	A
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The root nodes of a tree have height equal to:
((OPTION_A))	height of the tree
((OPTION_B))	zero

((OPTION_C))	one
((OPTION_D))	none of these
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>What is depth of following tree?</p>  <pre> graph TD 10((10)) --> 20((20)) 10 --> 60((60)) 20 --> 40((40)) 20 --> 30((30)) 60 --> 50((50)) 60 --> 90((90)) 50 --> 70((70)) 90 --> 80((80)) </pre>
((OPTION_A))	2
((OPTION_B))	4
((OPTION_C))	3
((OPTION_D))	1
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
-------------------------	---

((QUESTION))	A degree of any node in a binary tree is obtained from ---
((OPTION_A))	Number of leaf node
((OPTION_B))	Number of successor nodes
((OPTION_C))	Number of left subbranches
((OPTION_D))	Number of right subbranches
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	A sub-tree of binary search tree is ---
((OPTION_A))	Binary tree
((OPTION_B))	Balanced Tree
((OPTION_C))	Complete Binary tree
((OPTION_D))	Binary search tree
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	The degree of each node in general tree can be---
((OPTION_A))	At the most two
((OPTION_B))	Exactly two

((OPTION_C))	More than two
((OPTION_D))	Exactly three
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	An expression tree is always---
((OPTION_A))	A complete Binary tree
((OPTION_B))	Heap
((OPTION_C))	Binary Search tree
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In preorder sequence the root node is always at -----position
((OPTION_A))	First
((OPTION_B))	Last
((OPTION_C))	Mid

((OPTION_D))	Can't be predicted
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	In post order sequence the root node is always at -----position
((OPTION_A))	First
((OPTION_B))	Last
((OPTION_C))	Mid
((OPTION_D))	Can't be predicted
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Preorder traversal is nothing but
((OPTION_A))	DFS
((OPTION_B))	BFS
((OPTION_C))	Linear Order

((OPTION_D))	Topological Order
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	<p>Consider the following Fragment. Guess the traversal method.</p> <pre> Void Display(node* temp) { If(temp!=NULL) { Display(temp->left); Display(temp->right); Cout<<temp->data; } } </pre>
((OPTION_A))	Inorder
((OPTION_B))	Reorder
((OPTION_C))	Postorder
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	1
((QUESTION))	Consider the following Fragment. Guess the traversal method. <pre>Void Display(node* temp) { If(temp!=NULL) { Cout<<temp->data; Display(temp->left); Display(temp->right); } }</pre>
((OPTION_A))	Inorder
((OPTION_B))	Preorder
((OPTION_C))	Postorder
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

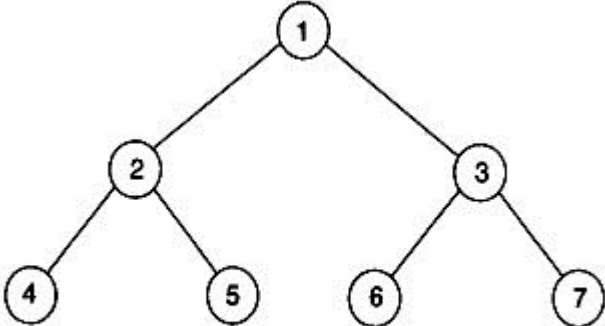
((MARKS)) (1/2/3...)	1
((QUESTION))	Consider the following Fragment. Guess the traversal method.

	<pre> Void Display(node* temp) { If(temp!=NULL) { Display(temp->left); Cout<<temp->data; Display(temp->right); } } </pre>
((OPTION_A))	Inorder
((OPTION_B))	Preorder
((OPTION_C))	Postorder
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Find Postfix Expression of following Infix Expression $A-B+C*D$
((OPTION_A))	$A-BC*D+$
((OPTION_B))	$AB-CD*+$
((OPTION_C))	$AB-+CD*$
((OPTION_D))	NONE
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Binary tree is implemented non-recursively. Complete following pseudo- for non-recursive in-order traversal

	<pre> NonrecInorder () { Stack s; TreeNode *currentNode = root; while (1) { while (currentNode) { (a) currentNode = currentNode -> leftChild; } if(s.IsEmpty()) return; currentNode = s.Top(); (b) Visit(currentNode); currentNode = currentNode -> rightChild; } } </pre>
((OPTION_A))	<pre> a: s.Push(currentNode); b: s.Pop(); </pre>
((OPTION_B))	<pre> a: s.Push(currentNode -> leftChild); b: s.Pop(currentNode); </pre>
((OPTION_C))	<pre> a: s.Push(currentNode -> rightChild); b: s.Pop(); </pre>
((OPTION_D))	<pre> a: s.Push(); b: s.Pop(currentNode); </pre>
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<p>If the post order traversal gives a b - c d * + then the label of the nodes 1, will be</p>  <pre> graph TD 1((1)) --- 2((2)) 1 --- 3((3)) 2 --- 4((4)) 2 --- 5((5)) 3 --- 6((6)) 3 --- 7((7)) </pre>
((OPTION_A))	+, -, *, a, b, c, d
((OPTION_B))	a, -, b, +, c, *, d
((OPTION_C))	a, b, c, d, -, *, +
((OPTION_D))	-, a, b, +, *, c, d
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Complete following pseudo code of level order traversal of binary tree

	<pre> Levelorder() { Queue q; // data of queue of type TreeNode TreeNode *currentNode = root; while(currentNode) { Visit(currentNode); if(currentNode -> leftChild) (a) if(currentNode -> rightChild) (b) if(c) return; currentNode = q.Front(); q.Pop (); } } </pre>
((OPTION_A))	a: q.Push(current->leftChild); b: q.Push(current->rightChild); c: q.IsFull()
((OPTION_B))	a: q.Push(current->leftChild); b: q.Push(current->rightChild); c: q.IsEmpty()
((OPTION_C))	a: q.Push(current->rightChild); b: q.Push(current->leftChild); c: q.IsEmpty()
((OPTION_D))	a: q.Push(current->rightChild); b: q.Push(current->leftChild); c: q.IsEmpty()
((CORRECT_CHOICE)) (A/B/C/D)	B

((EXPLANATI ON)) (OPTIONAL)	
-----------------------------------	--

((MARKS)) (1/2/3...)	2
((QUESTION))	It is necessary for Huffman encoding tree to be,
((OPTION_A))	AVL Tree
((OPTION_B))	Binary Tree
((OPTION_C))	Complete Binary Tree
((OPTION_D))	Binary Search Tree
((CORRECT_C HOICE)) (A/B/ C/D)	B
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> Algorithm doSomething(tree1, tree2) { if(tree1=NULL and tree2=NULL) ans=true; else if{ (tree1->data == tree2->data) {ans=doSomething(lchild(tree1),lchild(tree2)) If(ans) Ans= doSomething(Rchild(tree1),Rchild(tree2)) } return ans; } </pre>
((OPTION_A))	Checking leaf nodes
((OPTION_B))	Checking if two trees are equal
((OPTION_C))	Find left child
((OPTION_D))	Find right child
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> Int DoSomething(BstNode* root) {if(root->left == NULL) Return root->data; Else Return DoSomething(root->left) } </pre> <p>What does the above code do?</p>
((OPTION_A))	Returns any left child value
((OPTION_B))	Returns smallest value
((OPTION_C))	Returns intermediate value
((OPTION_D))	Returns root value
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> DoSomething(BstNode* root) {if(root!=NULL) cout<<root->data DoSomething(root->left) DoSomething(root->right) } </pre> <p>What does it do?</p>
((OPTION_A))	Returns level wise node values
((OPTION_B))	Returns root value
((OPTION_C))	Returns depth first searched values

((OPTION_D))	None
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	The min. number of nodes in a binary tree of depth d (root at level 0) is
((OPTION_A))	$2^d - 1$
((OPTION_B))	$2^d + 1 - 1$
((OPTION_C))	$d + 1$
((OPTION_D))	D
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	The post order traversal of a binary tree is DEBFCA. Find out the preorder traversal
((OPTION_A))	ABFCDE
((OPTION_B))	ADBFEC
((OPTION_C))	ABDECF
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	The height of a BST is given as h. Consider the height of the tree as the no. of edges in the longest path from root to the leaf. The maximum no. of nodes possible in the tree is?
((OPTION_A))	$2^{h-1} - 1$
((OPTION_B))	$2^{h+1} - 1$
((OPTION_C))	$2^h + 1$
((OPTION_D))	$2^{h-1} + 1$
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Suppose a binary tree is constructed with n nodes, such that each node has exactly either zero or two children. The maximum height of the tree will be?
((OPTION_A))	$(n+1)/2$
((OPTION_B))	$(n-1)/2$
((OPTION_C))	$n/2 - 1$
((OPTION_D))	$(n+1)/2 - 1$
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Suppose we have numbers between 1 and 1000 in a binary search tree and want to search for the number 363. Which of the following sequence could not be the sequence of the node examined?
((OPTION_A))	2, 252, 401, 398, 330, 344, 397, 363
((OPTION_B))	924, 220, 911, 244, 898, 258, 362, 363
((OPTION_C))	925, 202, 911, 240, 912, 245, 258, 363
((OPTION_D))	2, 399, 387, 219, 266, 382, 381, 278, 363
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	A binary search tree is formed from the sequence 6, 9, 1, 2, 7, 14, 12, 3, 8, 18. The minimum number of nodes required to be added in to this tree to
((OPTION_A))	3
((OPTION_B))	6
((OPTION_C))	8
((OPTION_D))	11
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>while (cur != NULL) { cout<< cur->data; // If this node is a thread node, then go to // inorder successor if (cur->rightThread) cur = cur->right; else cur = leftmost(cur->right); }</pre> <p>What does above code snippet do?</p>
((OPTION_A))	Inorder traversal in threaded binary tree
((OPTION_B))	BFS in threaded binary tree

((OPTION_C))	Insertion of a node in threaded binary tree
((OPTION_D))	Deletion of a node in threaded binary tree
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> if (root == NULL) return true; bool l = (root->left) ? getCountUtil(root->left, low, high, count) : true; bool r = (root->right) ? getCountUtil(root->right, low, high, count) : true; if (l && r && inRange(root, low, high)) { ++*count; return true; } return false; </pre> <p>What does above code snippet do?</p>
((OPTION_A))	Count leaf nodes given range
((OPTION_B))	Count internal nodes given range
((OPTION_C))	Count subtrees in given range
((OPTION_D))	Count branches given range
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION))	

ON)) (OPTIONAL)	
--------------------	--

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>if (node == NULL) return;</pre> <pre> DoSomething(node->left); DoSomething (node->right);</pre> <pre> Cout<<"\n Deleting node:"<<node->data; free(node);</pre> <p>What does above code snippet do?</p>
((OPTION_A))	Delete a node
((OPTION_B))	Delete entire tree
((OPTION_C))	Delete left subtree
((OPTION_D))	Delete only right subtree
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>if(node == NULL) return 0; if(node->left == NULL && node->right==NULL) return 1; else return ABC(node->left)+ ABC(node->right);</pre> <p>What does the function ABC do?</p>

((OPTION_A))	Count leaf nodes
((OPTION_B))	Count right subtree nodes only
((OPTION_C))	Count left subtree nodes only
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<p>pCrawl = root set initial stack element as NULL (sentinal) while(pCrawl is valid) stack.push(pCrawl) pCrawl = pCrawl.left while(pCrawl = stack.pop() is valid) stop if sufficient number of elements are popped. if(pCrawl.right is valid) pCrawl = pCrawl.right while(pCrawl is valid) stack.push(pCrawl) pCrawl = pCrawl.left What does above algorihm snippet do?</p>
((OPTION_A))	Finds kth smallest element in binary tree
((OPTION_B))	Traverses a tree
((OPTION_C))	BFS in tree
((OPTION_D))	DFS in tree
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATI	

ON)) (OPTIONAL)	
--------------------	--

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>int Tree(struct node* node) { if (node == NULL) return(true); if (node->left!=NULL && maxVal(node->left) > node->data) return(false); if (node->right!=NULL && minVal(node->right) < node->data) return(false); if (!isBST(node->left) !isBST(node->right)) return(false); return(true); }</pre> <p>What does Tree() do?</p>
((OPTION_A))	Checks whether tree has left subtree or not
((OPTION_B))	Checks whether tree has right subtree or not
((OPTION_C))	Checks whether tree is BST or not
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> void print (struct node* root) { int h = height(root); int i; for (i=1; i<=h; i++) print1 (root, i); } void print1(struct node* root, int level) { if (root == NULL) return; if (level == 1) printf("%d ", root->data); else if (level > 1) { print1(root->left, level-1); print1(root->right, level-1); } } </pre> <p>What does print () do?</p>
((OPTION_A))	Inorder traversal using stack
((OPTION_B))	Traverses a tree level wise
((OPTION_C))	Generally traverses a tree in random way
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>int find(node* a, node* b) { if (a==NULL && b==NULL) return 1; if (a!=NULL && b!=NULL) { return (a->data == b->data && find(a->left, b->left) && find(a->right, b->right)); } return 0; }</pre> <p>What does above code snippet do?</p>
((OPTION_A))	Finds if two trees are identical
((OPTION_B))	Finds if a tree and its mirror are identical
((OPTION_C))	Finds the existence of left and right subtrees
((OPTION_D))	None of these
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>Void func(node* node1) { if (node1==NULL) return; else { node* temp; func(node1->left); func(node1->right); } }</pre>

	<pre> temp = node1->left; node1->left = node1->right; node1->right = temp; } </pre>
((OPTION_A))	Mirror a tree
((OPTION_B))	Find duplicate of a tree
((OPTION_C))	Finds right subtree
((OPTION_D))	Finds left subtree
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> int func(struct node* node) { if (node==NULL) return 0; else { /* compute the depth of each subtree */ int l = func(node->left); int r = func(node->right); /* use the larger one */ if (l > r) return(l+1); else return(r+1); } } </pre> <p>What does func() do?</p>

((OPTION_A))	Finds height of a tree
((OPTION_B))	Finds levels in a tree
((OPTION_C))	Finds number of nodes in a tree
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>int func(node* node) { node* current = node; /* loop down to find the leftmost leaf */ while (current->left != NULL) { current = current->left; } return(current->data); }</pre> <p>What does func() do?</p>
((OPTION_A))	Finds minimum element in a tree
((OPTION_B))	Finds left leaves in a tree
((OPTION_C))	Finds number of nodes in a tree
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> node *abc(node* root, int n1, int n2) { if (root == NULL) return NULL; // If both n1 and n2 are smaller than root, then LCA lies in left if (root->data > n1 && root->data > n2) return abc(root->left, n1, n2); // If both n1 and n2 are greater than root, then LCA lies in right if (root->data < n1 && root->data < n2) return abc(root->right, n1, n2); return root; } </pre> <p>What does abc() do?</p>
((OPTION_A))	Finds minimum element in a tree
((OPTION_B))	Finds left leaves in a tree
((OPTION_C))	Finds lowest common ancestor
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>bool func(int pre[], int size) { int nextDiff, lastDiff; for (int i=0; i<size-1; i++) { nextDiff = pre[i] - pre[i+1]; lastDiff = pre[i] - pre[size-1]; if (nextDiff*lastDiff < 0) return false;; } return true; }</pre> <p>What does func() do?</p>
((OPTION_A))	Finds minimum element in a tree
((OPTION_B))	Finds left leaves in a tree
((OPTION_C))	Finds number of nodes in a tree with 1 child
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>{ if(n->right != NULL) return minValue(n->right); struct node *p = n->parent; while(p != NULL && n == p->right) { n = p; p = p->parent; } }</pre>

	<pre> } return p; } </pre> <p>What does above do?</p>
((OPTION_A))	Finds minimum element in a tree
((OPTION_B))	Finds inorder successor of a node
((OPTION_C))	Finds number of nodes in a tree with 1 child
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> {if (root == NULL) return ; if (root->key == key) { if (root->left != NULL) { Node* tmp = root->left; while (tmp->right) tmp = tmp->right; pre = tmp ; } if (root->right != NULL) { Node* tmp = root->right ; while (tmp->left) tmp = tmp->left ; suc = tmp ; } return ; return true; } </pre> <p>What does above code do?</p>

((OPTION_A))	Finds minimum and maximum element in a tree
((OPTION_B))	Finds predecessor and successor of a node in a tree
((OPTION_C))	Finds number of nodes in a tree with 1 child
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> {if (root == NULL) return ; if (root->key == key) { if (root->left != NULL) { Node* tmp = root->left; while (tmp->right) tmp = tmp->right; pre = tmp ; } if (root->right != NULL) { Node* tmp = root->right ; while (tmp->left) tmp = tmp->left ; suc = tmp ; } return ; } return true; } </pre> <p>What does above code do?</p>
((OPTION_A))	Finds minimum and maximum element in a tree
((OPTION_B))	Finds predecessor and successor of a node in a tree
((OPTION_C))	Finds number of nodes in a tree with 1 child

((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>int abc(struct node *root) { if (isBST(root)) return size(root); else return max(abc (root->left), abc (root->right)); }</pre> <p>What does abc() do?</p>
((OPTION_A))	Finds minimum subtree in a tree
((OPTION_B))	Finds largest BST subtree
((OPTION_C))	Finds left and right subtrees
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre>void abc(Node *root, int &c) { // Base cases, the second condition is important to // avoid unnecessary recursive calls</pre>

	<pre> if (root == NULL c >= 2) return; // Follow reverse inorder traversal so that the // largest element is visited first abc (root->right, c); // Increment count of visited nodes c++; // If c becomes k now, then this is the 2nd largest if (c == 2) { cout << "2nd largest element is " << root->key << endl; return; } // Recur for left subtree abc (root->left, c); } </pre> <p>What does abc() do?</p>
((OPTION_A))	Finds minimum subtree in a tree
((OPTION_B))	Finds largest element
((OPTION_C))	Finds 2 nd largest element
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	C
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> void abc (struct node *root) { // base case: tree is empty if(root == NULL) return; int n = countNodes (root); int *arr = new int[n]; int i = 0; storeInorder (root, arr, &i); } </pre>

	<pre> qsort (arr, n, sizeof(arr[0]), compare); i = 0; arrayToBST (arr, root, &i); delete [] arr; } </pre> <p>What does abc() do?</p>
((OPTION_A))	Converts a binary tree to BST
((OPTION_B))	Finds largest BST subtree
((OPTION_C))	Finds left and right subtrees
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> class Node { int data; Node *left, *right; bool t; } </pre> <p>Which data structure does the above class define?</p>
((OPTION_A))	Threaded binary tree
((OPTION_B))	Single threaded binary tree
((OPTION_C))	Double threaded binary tree
((OPTION_D))	BST
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	In k ary tree what is relation between leaf nodes and internal nodes? (L= leaf nodes and I =internal nodes)
((OPTION_A))	$L = (k - 1) * I + 1$
((OPTION_B))	$L = K * I + 1$
((OPTION_C))	$L = K - I$
((OPTION_D))	$L = K + 1 * I$
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<pre> int abc(node* node) { if (node==NULL) return 0; else return(abc(node->left) + 1 + abc(node->right)); } </pre>
((OPTION_A))	Finds number of nodes in tree
((OPTION_B))	Finds number of nodes in left sub tree
((OPTION_C))	Finds number of nodes in right sub tree
((OPTION_D))	None of the above
((CORRECT_CHOICE)) (A/B/C/D)	A

((EXPLANATION)) (OPTIONAL)	
-------------------------------	--

((MARKS)) (1/2/3...)	2
((QUESTION))	In a binary tree with n nodes, every node has an odd number of descendants. Every node is considered to be its own descendant. What is the number of nodes in the tree that have exactly one child?
((OPTION_A))	0
((OPTION_B))	1
((OPTION_C))	$(n-1)/2$
((OPTION_D))	$n-1$
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Consider a binary tree T that has 200 leaf nodes. Then, the number of nodes in T that have exactly two children are _____.
((OPTION_A))	199
((OPTION_B))	200
((OPTION_C))	Any number between 0 and 199
((OPTION_D))	Any number between 100 and 200
((CORRECT_CHOICE)) (A/B/C/D)	A

C/D)	
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	A binary tree T has 20 leaves. The number of nodes in T having two children is
((OPTION_A))	18
((OPTION_B))	19
((OPTION_C))	17
((OPTION_D))	Any number between 10 and 20
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	In a complete k-ary tree , every internal node has exactly k children. The number of leaves in such a tree with n internal nodes is
((OPTION_A))	Nk
((OPTION_B))	$(n - 1)k + 1$
((OPTION_C))	$n(k - 1) + 1$
((OPTION_D))	$n(k - 1)$
((CORRECT_CHOICE))	C

HOICE)) (A/B/ C/D)	
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	A complete n-ary tree is a tree in which each node has n children or no children. Let I be the number of internal nodes and L be the number of leaves in a complete n-ary tree. If $L = 41$, and $I = 10$, what is the value of n?
((OPTION_A))	3
((OPTION_B))	4
((OPTION_C))	5
((OPTION_D))	6
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	<p>The hight of a tree is defined as the number of edges on the longest path in the tree.The function shown in the pseudocode below is invoked as height(root) to compute the height of a binary tree rooted at the tree pointer root.</p> <pre> int height { if (n= =NULL) return -1; If(n-> left= =NULL) If(n-> right= =NULL)) return 0;h1 = height (n -> left); Else return B1; Else { h1 = height (n -> left); If (n -> right = NULL) return (1 + h1); Else { h2 = height (n ~ right); Return B2; } } } </pre> <p>The appropriate expression for the two boxes B1 and B2 are_____.</p>
((OPTION_A))	B1:(1 + height(n-> right)) ,B2: (1 + max (h1,h2))
((OPTION_B))	B1:(1 + height(n-> right)) ,B2: (1 + max (h1,h2))
((OPTION_C))	B1:(height(n-> right)) ,B2: (max (h1,h2))
((OPTION_D))	B1:(1 + height(n-> right)) ,B2: (max (h1,h2))
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Find Huffman Code for node E. If Weights of nodes are as follows: A=10, B=3,C=4,D=15,E=2.
((OPTION_A))	0001
((OPTION_B))	1010
((OPTION_C))	1000
((OPTION_D))	1000
((CORRECT_C HOICE)) (A/B/ C/D)	C
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Post fix expression of an expression tree is $ab+cd-*$. Find inorder traversal of it
((OPTION_A))	$a+b-c*d$
((OPTION_B))	$ab+c*d-$
((OPTION_C))	$a-cd+*$
((OPTION_D))	None
((CORRECT_C HOICE)) (A/B/ C/D)	A
((EXPLANATI ON)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	DFS of a tree is 10,4,8,6,5,7,9,12,11.Find BFS.
((OPTION_A))	10 4 12 3 8 11 6 9 5 7
((OPTION_B))	10 12 8 11 6 5 9 7 4 3
((OPTION_C))	10 8 12 11 5 9 6 7 3 4
((OPTION_D))	None
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	DFS of a tree is : 10 5 12 4 7 6 9 8. If 7 is to be deleted it will be replaced by which node?
((OPTION_A))	10
((OPTION_B))	4
((OPTION_C))	5
((OPTION_D))	8
((CORRECT_CHOICE)) (A/B/C/D)	D
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	DFS of a tree is : 10 5 12 4 7 6 9 8. If 10 is to be deleted it will be replaced by which node?
((OPTION_A))	7
((OPTION_B))	12
((OPTION_C))	5
((OPTION_D))	8
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	DFS of a tree is : 10 5 12 4 7 6 9 8. If 4 is to be deleted it will be replaced by which node?
((OPTION_A))	7
((OPTION_B))	12
((OPTION_C))	5
((OPTION_D))	8
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Inorder traversal of a tree is: 40 20 30 10 70 50 60 80 90. When 20 is deleted with which node is it replaced?
((OPTION_A))	30
((OPTION_B))	40
((OPTION_C))	10
((OPTION_D))	50
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Tree has 7 nodes? How many orders of traversals are possible?
((OPTION_A))	256
((OPTION_B))	128
((OPTION_C))	7
((OPTION_D))	70
((CORRECT_CHOICE)) (A/B/C/D)	B
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	Following nodes are inserted in BST:10 8 15 12 13 7 9. What is height of the tree?
((OPTION_A))	4
((OPTION_B))	5
((OPTION_C))	3
((OPTION_D))	2
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

((MARKS)) (1/2/3...)	2
((QUESTION))	How many nodes a complete binary tree of level 4 has?
((OPTION_A))	15
((OPTION_B))	13
((OPTION_C))	11
((OPTION_D))	9
((CORRECT_CHOICE)) (A/B/C/D)	A
((EXPLANATION)) (OPTIONAL)	

Id	
Question	A Complete Binary tree with the property that the value at each node is at least as large as the value at its children is called_____.
A	Binary Search Tree
B	AVL Tree
C	Completely balance tree
D	Heap
Answer	D
Marks	2
Unit	1

Id	
Question	A fully binary tree with n leaves contains.....
A	n nodes
B	$\text{Log}_2 n$ nodes
C	$2n-1$ nodes
D	2^n nodes
Answer	C
Marks	2
Unit	1

Id	
Question	Traversing a Binary tree first root and then left and right sub-trees called traversal.
A	Postorder
B	Preorder
C	Inorder
D	None of this
Answer	B
Marks	1
Unit	1

Id	
Question	The operation of processing each element in the list is known as
A	Sorting
B	Merging
C	Inserting
D	Traversal
Answer	D
Marks	1
Unit	1

Id	
Question	The no of external nodes in a full binary tree with n internal nodes is ?
A	n
B	$n+1$
C	$2n$
D	$2n + 1$
Answer	B
Marks	1
Unit	1

Id	
Question	Linked representation of binary tree needs parallel arrays.
A	4
B	2
C	3
D	1
Answer	C
Marks	2
Unit	1

Id	
Question	Which type of traversal of binary search tree outputs the value in sorted order?
A	Pre-order
B	In-order
C	Post-order
D	None
Answer	B
Marks	1
Unit	1

Id	
Question	Which indicates pre-order traversal?
A	Left sub-tree, Right sub-tree and root
B	Right sub-tree, Left sub-tree and root
C	Root, Left sub-tree, Right sub-tree
D	Right sub-tree, root, Left sub-tree
Answer	C
Marks	1
Unit	1

Id	
Question	A terminal node in a binary tree is called
A	Root
B	Leaf
C	Child
D	Branch
Answer	B
Marks	1
Unit	1

Id	
Question	The post order traversal of a binary tree is DEBFCA. Find out the pre order traversal
A	ABFCDE
B	ADBFEC
C	ABDECF
D	ABDCEF
Answer	C
Marks	2
Unit	1

Id	
Question	Pre orders traversal recursively
A	<pre> void preorder (Node * Root) { if (Root != NULL) { printf ("%d\n",Root → Info); preorder(Root → L child); preorder(Root → R child); } } </pre>
B	<pre> void preorder (Node * Root) { if (Root != NULL) { preorder(Root → L child); printf ("%d\n",Root → Info); preorder(Root → R child); } } </pre>
C	<pre> void preorder (Node * Root) { if (Root != NULL) { preorder(Root → L child); preorder(Root → R child); printf ("%d\n",Root → Info); } } </pre>
D	<pre> void preorder (Node * Root) { if (Root = NULL) { preorder(Root → L child); preorder(Root → R child); printf ("%d\n",Root → Info); } } </pre>
Answer	A
Marks	2
Unit	1

Id	
Question	In order traversal recursively
A	<pre>void inorder (NODE *Root) { If (Root == NULL) { inorder(Root → L child); printf ("%d\n",Root → info); inorder(Root → R child); } }</pre>
B	<pre>void inorder (NODE *Root) { If (Root != NULL) { inorder(Root → L child); printf ("%d\n",Root → info); inorder(Root → R child); } }</pre>
C	<pre>void inorder (NODE *Root) { If (Root = NULL) { inorder(Root → L child); printf ("%d\n",Root → info); inorder(Root → R child); } }</pre>
D	None of these
Answer	B
Marks	2
Unit	1

Id	
Question	Post order traversal recursively
A	<pre> void postorder (NODE *Root) { If (Root = NULL) { postorder(Root → Lchild); postorder(Root → Rchild); printf ("%d\n",Root → info); } } </pre>
B	<pre> void postorder (NODE *Root) { If (Root = NULL) { printf ("%d\n",Root → info); postorder(Root → Lchild); postorder(Root → Rchild); } } </pre>
C	<pre> void postorder (NODE *Root) { If (Root != NULL) { postorder(Root → Lchild); postorder(Root → Rchild); printf ("%d\n",Root → info); } } </pre>
D	<pre> void postorder (NODE *Root) { If (Root = NULL) { postorder(Root → Lchild); printf ("%d\n",Root → info); postorder(Root → Rchild); } } </pre>
Answer	C
Marks	2
Unit	1

Id	
Question	Which of the following statements is false
A	Every tree is a bipartite graph
B	A tree with n nodes contains $n-1$ edges
C	A tree is a connected graph
D	A tree contains a cycle
Answer	D
Marks	1
Unit	1

Id	
Question	If the binary tree has 50 nodes then the number of edges are
A	51
B	55
C	49
D	50
Answer	C
Marks	1
Unit	1

Id	
Question	While converting the binary tree to general tree the links are added from each node to
A	Its immediate right child
B	Its immediate left child
C	Its leaf node
D	None of these
Answer	A
Marks	2
Unit	1

Id	
Question	If a bottom level of a binary tree is not filled completely then it depicts that the tree is not
A	Threaded binary tree
B	Completely binary tree
C	Perfect binary tree
D	None of these
Answer	B
Marks	
Unit	

Id	
Question	By using _____ the recursive method of traversing a tree is avoided.
A	Binary tree
B	Expression tree
C	Threaded binary tree
D	Balanced tree
Answer	C
Marks	1
Unit	1

Id	
Question	What are total number of leaf nodes in a complete binary tree with depth 3
A	17
B	8
C	9
D	16
Answer	B
Marks	2
Unit	1

Id	
Question	The inorder and preorder traversal of a binary tree are d b e a f c g and a b d e c f g respectively. The postorder traversal of the binary tree is
A	d e b f g c a
B	e d b g f c a
C	e d b f g c a
D	d e f g b c a
Answer	A
Marks	2
Unit	1

Id	
Question	A Threaded Binary Tree is a binary tree in which every node that does not have a right child has a THREAD (in actual sense, a Link) _____Successor.
A	Preorder
B	Inorder
C	Postorder
D	Levelorder
Answer	B
Marks	2
Unit	1

Id	
Question	The in-order traversal of binary tree produced the sequence HFIEJGZ, and the post-order traversal of the same tree produced the sequence HIFJZGE. What will be the inorder of the left sub tree of the given tree?
A	HFI
B	HEI
C	HIF
D	HIE
Answer	A
Marks	2
Unit	1

Id	
Question	The in-order traversal of binary tree produced the sequence HFIEJGZ, and the post-order traversal of the same tree produced the sequence HIFJZGE. What will be the preorder of the binary tree?
A	EFGHIJZ
B	EFHIGJZ
C	EFIHJGZ
D	EFHIJGZ
Answer	B
Marks	2
Unit	1

Id	
Question	The in-order traversal of binary tree produced the sequence CBDAFGE, and the pre-order traversal of the same tree produced the sequence ABCDEFG. What will be the preorder of the right sub tree of given binary tree?
A	EFG
B	FGE
C	GEF
D	GFE
Answer	A
Marks	2
Unit	1

Id	
Question	Which of the following number of nodes can have a full binary tree
A	8
B	13
C	15
D	14
Answer	C
Marks	2
Unit	1

Id	
Question	A binary tree in which the descendants points to its ancestor is called_____
A	Binary search tree
B	Complete binary tree
C	Threaded binary tree
D	Balanced tree
Answer	C
Marks	2
Unit	1

Id	
Question	A binary search tree contains the values - 1, 2, 3, 4, 5, 6, 7 and 8. The tree is traversed in preorder and the values are printed out. Which of the following sequences is a valid output?
A	5 3 2 4 1 6 7 8
B	5 3 1 2 4 6 4 9 7
C	5 3 1 2 4 7 8 6
D	5 3 1 2 4 7 6 8
Answer	D
Marks	2
Unit	1

Id	
Question	A binary tree is generated by inserting in order the following integers:50, 15, 62, 5, 20,58, 91, 3,8,37, 60, 24 The number of nodes in the left and right of the root respectively is
A	(4,7)
B	(6,3)
C	(7,5)
D	(3,6)
Answer	C
Marks	2
Unit	1

Id	
Question	Consider the binary_tree_node t. Which expression indicates that t represents an empty tree
A	(t == NULL)
B	(t->data() == 0)
C	(t->data() == NULL)
D	((t->left() == NULL) && (t->right() == NULL))
Answer	A
Marks	1
Unit	1

Id	
Question	Select the one true statement
A	Every binary tree is either complete or full
B	Every complete binary tree is also a full binary tree
C	Every full binary tree is also a complete binary tree.
D	No binary tree is both complete and full.
Answer	C
Marks	2
Unit	1

Id	
Question	Suppose T is a binary tree with 14 nodes. What is the minimum possible depth of T
A	0
B	3
C	4
D	5
Answer	B
Marks	2
Unit	1

Id	
Question	Select the one FALSE statement about binary trees
A	Every binary tree has at least one node.
B	Every non-empty tree has exactly one root node.
C	Every node has at most two children.
D	Every non-root node has exactly one parent
Answer	A
Marks	2
Unit	1

Id	
Question	Consider the node of a complete binary tree whose value is stored in <code>data[i]</code> for an array implementation. If this node has a right child, where will the right child's value be stored
A	<code>data[i+1]</code>
B	<code>data[i+2]</code>
C	<code>data[2*i + 1]</code>
D	<code>data[2*i + 2]</code>
Answer	B
Marks	2
Unit	1

Id	
Question	A full binary tree with $2n+1$ nodes
A	n leaf nodes
B	n non-leaf nodes
C	n-1 leaf nodes
D	n-1 non-leaf nodes
Answer	B
Marks	2
Unit	1

Id	
Question	A BST is traversed in the following order recursively: Right, root, left The output sequence will be
A	Ascending order
B	Descending sequence
C	Can't say
D	None of the above
Answer	B
Marks	2
Unit	1

Id	
Question	What is the post order traversal of a tree if the preorder traversal is 14 2 1 3 11 10 7 30 40 & inorder traversal is 1 2 3 14 7 10 11 40 30
A	40 30 14 11 10 7 3 2 1
B	1 2 3 40 30 10 7 11 14
C	1 3 2 7 10 40 30 11 14
D	None of these
Answer	C
Marks	2
Unit	1

Id	
Question	The number of leaf nodes in a complete binary tree of depth d is____
A	$2d$
B	$2d-1+1$
C	$2d+1+1$
D	$2d+1$
Answer	A
Marks	1
Unit	1

Id	
Question	A B-tree of minimum degree t can maximum _____ pointers in a node.
A	$t-1$
B	$2t-1$
C	$2t$
D	t
Answer	D
Marks	1
Unit	1

Id	
Question	Binary trees with threads are called as.....
A	Threaded trees
B	Special pointer trees
C	Special trees
D	Pointer trees
Answer	A
Marks	1
Unit	1

Id	
Question	In Binary trees nodes with no successor are called
A	End nodes
B	Terminal nodes
C	Final nodes
D	Last nodes
Answer	B
Marks	1
Unit	1

Id	
Question	Tree
A	Is a bipartite graph
B	With n node contains n-1 edges
C	Is a connected graph
D	All of these
Answer	D
Marks	1
Unit	1

Id	
Question	A graph in which all nodes are of equal degrees is known as
A	Complete graph
B	Regular graph
C	Non regular graph
D	Multi graph
Answer	B
Marks	1
Unit	1

Id	
Question	The basic idea behind Huffman coding is to
A	Compress data by using fewer bits to encode fewer frequently occurring characters
B	Expand data by using fewer bits to encode more frequently occurring characters
C	Compress data by using fewer bits to encode more frequently occurring characters
D	Compress data by using more bits to encode more frequently occurring characters
Answer	C
Marks	1
Unit	1

Id	
Question	An alphabet consist of the letters A, B, C and D. The probability of occurrence is $P(A) = 0.4$, $P(B) = 0.1$, $P(C) = 0.2$ and $P(D) = 0.3$. The Huffman code is
A	$A = 01$ $B = 111$ $C = 110$ $D = 10$
B	$A = 0$ $B = 111$ $C = 110$ $D = 10$
C	$A = 0$ $B = 111$ $C = 11$ $D = 101$
D	$A = 0$ $B = 11$ $C = 10$ $D = 111$
Answer	B
Marks	2
Unit	1

Id	
Question	Huffman coding is an encoding algorithm used for
A	broadband systems
B	files greater than 1 Mbit
C	lossless data compression
D	lossy data compression
Answer	C
Marks	1
Unit	1

Id	
Question	A Huffman encoder takes a set of characters with fixed length and produces a set of characters of
A	constant length
B	fixed length
C	variable length
D	random length
Answer	C
Marks	1
Unit	1

Id	
Question	A Huffman code: A = 1, B = 000, C = 001, D = 01 $P(A) = 0.4$, $P(B) = 0.1$, $P(C) = 0.2$, $P(D) = 0.3$ The average number of bits per letter is
A	8.0 bit
B	2.0 bit
C	2.1 bit
D	1.9 bit
Answer	D
Marks	2
Unit	1

Id	
Question	what is an optimal Huffman code for alphabet a of the following set of frequencies a: 05, b:48, c:07, d:17, e:10, f:13
A	1010
B	0101
C	1001
D	1100
Answer	A
Marks	2
Unit	1

Id	
Question	what is an optimal Huffman code for alphabet b of the following set of frequencies a: 45, b:13, c:12, d:16, e:9, f:5
A	100
B	111
C	001
D	101
Answer	D
Marks	2
Unit	1

Id	
Question	What is an optimal Huffman code for alphabet e of the following set of frequencies a: 29, b:25, c:20, d:12, e:05, f:09
A	1000
B	1110
C	0010
D	1011
Answer	B
Marks	2
Unit	1

Id	
Question	In delete operation of BST, we need inorder successor (or predecessor) of a node when the node to be deleted has both left and right child as non-empty. Which of the following is true about inorder successor needed in delete operation?
A	Inorder successor is always either a leaf node or a node with empty left child
B	Inorder Successor is always a leaf node
C	Inorder successor may be an ancestor of the node
D	Inorder successor is always either a leaf node or a node with empty right child
Answer	A
Marks	2
Unit	1

Id	
Question	Suppose the numbers 7, 5, 1, 8, 3, 6, 0, 9, 4, 2 are inserted in that order into an initially empty binary search tree. The binary search tree uses the usual ordering on natural numbers. What is the in-order traversal sequence of the resultant tree?
A	9 8 6 4 2 3 0 1 5 7
B	0 2 4 3 1 6 5 9 8 7
C	7 5 1 0 3 2 4 6 8 9
D	0 1 2 3 4 5 6 7 8 9
Answer	D
Marks	2
Unit	1

Id	
Question	Which of the following traversals is sufficient to construct BST from given traversals 1) Inorder 2) Preorder 3) Postorder
A	Any one of the given three traversals is sufficient
B	Either 2 or 3 is sufficient
C	1 and 3
D	2 and 3
Answer	B
Marks	2
Unit	1