Binary Search Tree MCQ Quiz - Objective Question with Answer for Binary Search Tree - Download Free PDF

Last updated on Feb 21, 2025

Binary Search Tree MCQs are crucial for assessing one's understanding of this data structure used for efficient searching and sorting operations. Binary search trees organize data in a hierarchical structure, enabling fast retrieval and insertion. Binary Search Tree MCQ evaluate learners' knowledge of tree traversal, node insertion and deletion, tree balancing, and operations on binary search trees. By answering Binary Search Tree MCQs, individuals can enhance their understanding of binary search tree properties, algorithms, and their applications in data processing, databases, and algorithm design.

Latest Binary Search Tree MCQ Objective Questions

Binary Search Tree Question 1:

Consider a completely skewed (left/right) binary search tree with n elements. What is the worst case time complexity of searching an element in this tree?

1. O(n)
2. O(1)
3. O(log n)
4. O(nlogn)

Answer (Detailed Solution Below)

Option 1 : O(n)

Binary Search Tree Question 1 Detailed Solution

The correct answer is **O(n).**

Key Points

- **A binary search tree (BST) is a binary tree where each node has a value greater than all the values in its left subtree and less than all the values in its right subtree.**
- In a balanced BST, the time complexity for searching is O(log n) due to the tree's height being log(n).
- However, in a completely skewed (left or right) BST, the tree essentially behaves like a linked list.
- This means each node only has one child, and the tree's height becomes n (number of nodes).
- Therefore, the worst-case time complexity of searching an element in a completely skewed BST is O(n) because you may need to traverse all the nodes.

Important Points

- In a balanced BST, operations like insertion, deletion, and search have average time complexities of O(log n).
- In a completely skewed BST, these operations degrade to O(n) in the worst case.

Additional Information

- **Self-balancing BSTs like AVL trees and Red-Black trees maintain their height close to log(n), ensuring efficient operations.**
- **Understanding the structure and properties of different types of BSTs is crucial for optimizing search operations in various applications.**

Binary Search Tree Question 2:

_________ is a Self Balancing binary search tree, where the path from the root to the furthest leaf is no more than twice as long as the path from the root to nearest leaf.

1. Expression tree
2. Game tree
3. Red-Black tree
4. Threaded tree

Answer (Detailed Solution Below)

Option 3 : Red-Black tree

Binary Search Tree Question 2 Detailed Solution

The correct answer is **Red-Black tree**

Key Points

- **Red-Black Tree**: A Red-Black tree is a type of self-balancing binary search tree.
- It maintains balance by ensuring that the longest path from the root to a leaf is no more than twice as long as the shortest path from the root to any leaf.
- This balancing property ensures that the tree remains approximately balanced, allowing operations such as insertion, deletion, and lookup to be performed in **O(logn)** time.

Additional Information

- **Expression Tree**: A binary tree used to represent expressions, not necessarily balanced.
- **Game Tree**: A tree representation of the possible moves in a game, not a self-balancing binary search tree.
- **Threaded Tree:** A binary tree in which null pointers are made to point to the in-order predecessor or successor, not specifically self-balancing.

Binary Search Tree Question 3:

A binary search tree T contains n distinct elements. What is the time complexity of picking an element in T that is smaller than the maximum element in T?

1. $\Theta(1)$
2. $\Theta(n \log n)$
3. $\Theta(\log n)$
4. More than one of the above
5. None of the above

Answer (Detailed Solution Below)

Option 1 : $\Theta(1)$

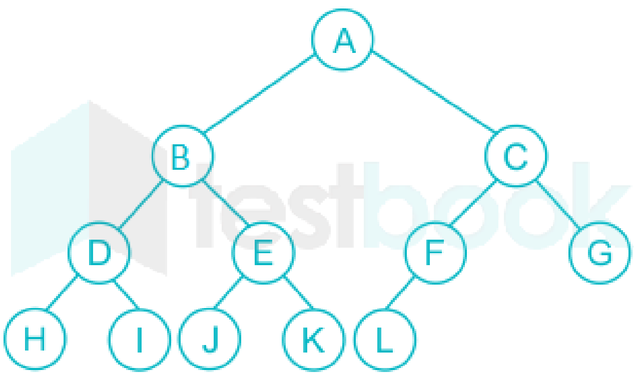
Binary Search Tree Question 3 Detailed Solution

Explanation:

- If an element in a binary search tree is smaller than any other element in the binary search tree then it is smaller than the maximum element.
- All the elements in the binary search tree is distinct.
- Compare only two elements in the binary search tree to find such elements.

Binary Search Tree Question 4:

Which of the following statements about the following binary tree is FALSE?



1. Nodes 'J' and 'K' are siblings.
2. Node 'B' is the ancestor of node 'J'
3. It is a binary search tree.
4. It is a complete binary tree.
5. None of the above

Answer (Detailed Solution Below)

Binary Search Tree Question 4 Detailed Solution

Concept:

Binary search tree: A BST is a tree in which all the nodes follow the two properties.

1) The left sub tree of a node has a key less than or equal to its parent node's key.

2) The right sub tree of a node has a key greater than its parent's key.

Complete binary tree: A complete binary tree is a binary tree in which every level except possibly the last level is completely filled and all nodes are as left as possible.

Explanation:

In this tree, it is clearly showing that node J and K are siblings.

Also, given tree is satisfying the property of a complete binary tree.

But it is not following the property of binary search tree. So, option 3) it is a binary search tree is incorrect here.

Binary Search Tree Question 5:

Which of the followings are true for a complete binary tree ?

A. It has always odd number of vertices.

B. With i internal vertices, it has i + 1 leaves.

C. With ℓ leaves it has ℓ - 1 vertices.

D. With 2n - 1 vertices, it has n leaves.

Choose the correct answer from the options given below :

2. B, C Only

3. A, D Only

4. A, B, C, D

Answer (Detailed Solution Below)

Option 2 : B, C Only

Binary Search Tree Question 5 Detailed Solution

The correct answer is **2) B, C Only**.

Key Points

- **Statement B:** In a complete binary tree, with i internal vertices, it has i + 1 leaves. This is true because in a complete binary tree, every internal node has exactly two children. Therefore, if there are i internal nodes, there will be i + 1 leaves.
- **Statement C:** In a complete binary tree, with ℓ leaves it has ℓ - 1 internal vertices. This is true because in a complete binary tree, the number of internal vertices is always one less than the number of leaves.

Additional Information

- **Statement A:** It has always odd number of vertices. This is incorrect. A complete binary tree can have an odd or even number of vertices.
- **Statement D:** With 2n - 1 vertices, it has n leaves. This is incorrect. The correct relationship between the number of vertices and leaves in a complete binary tree does not follow this formula.

testbook English Get Started

Top Binary Search Tree MCQ Objective Questions

Binary Search Tree Question 6

Download Solution PDF

The preorder traversal sequence of a binary search tree is 30, 20, 10, 15, 25, 23, 39, 35, 42. Which one of the following is the postorder traversal sequence of the same tree?

1. 10, 20, 15, 23, 25, 35, 42, 39, 30
2. 15, 10, 25, 23, 20, 42, 35, 39, 30
3. 15, 20, 10, 23, 25, 42, 35, 39, 30
4. 15, 10, 23, 25, 20, 35, 42, 39, 30

Answer (Detailed Solution Below)

Option 4 : 15, 10, 23, 25, 20, 35, 42, 39, 30

Binary Search Tree Question 6 Detailed Solution

Download Solution PDF

The correct answer is "**option 4".**

CONCEPT:

A Binary Search Tree (BST) is also known as an **ordered tree** or **sorted binary tree.**

It is a binary tree with the following properties:

1. The **left sub-tree** of a node contains only nodes with key-value **lesser** than the node's key value.

2. The **right subtree** of a node contains only nodes with a key-value **greater** than the node's key value.

There are **three types** of traversal:

1. **In-order traversal:** In this traversal, the first left node will traverse, the root node then the right node will get traversed.

3. **Post-order traversal**: In this traversal, the First left node will traverse, the right node then the root node will get traversed.

The in-order traversal of the Binary search tree always returns key values in ascending order.

EXPLANATION:

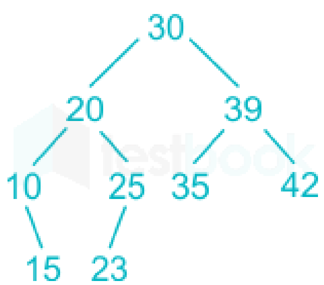
The **pre-order traversal** of given BST is:

30, 20, 10, 15, 25, 23, 39, 35, 42.

So, the **In-order traversal** of the BST is:

10, 15, 20, 23, 25, 30, 35, 39, 42.

The **Binary Search Tree** is:



So the post-order traversal of the tree is:

15, 10, 23, 25, 20, 35, 42, 39, 30

Hence, the correct answer is "option 4".

Binary Search Tree Question 7

II. 5, 8, 9, 12, 10, 15, 25

III. 2, 7, 10, 8, 14, 16, 20

IV. 4, 6, 7, 9 18, 20, 25

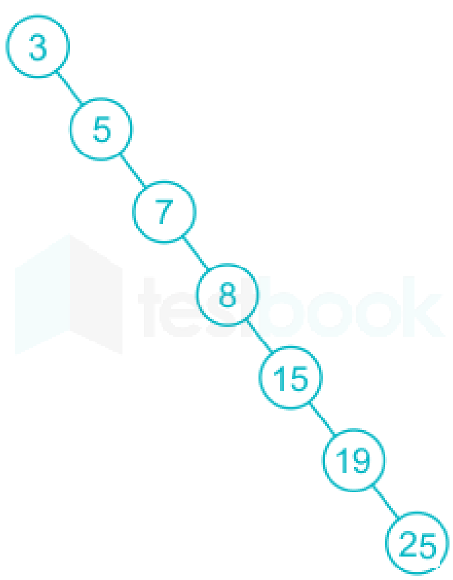
1. I and IV only
2. II and III only
3. II and IV only
4. II only

Answer (Detailed Solution Below)

Option 1 : I and IV only

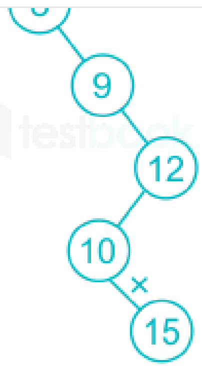
Binary Search Tree Question 7 Detailed Solution

Statement I: 3, 5, 7, 8, 15, 19, 25



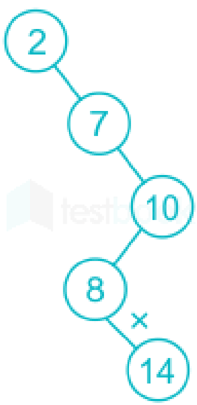
It doesn't violate Binary search tree property and hence it is the correct order of traversal.

Statement II: 5, 8, 9, 12, 10, 15, 25



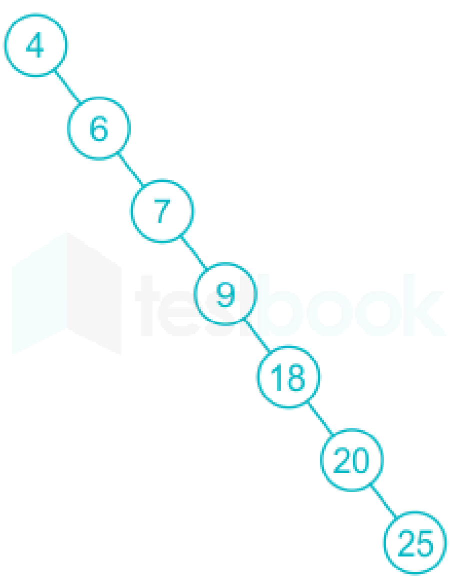
15 is left of 12 which violates binary search tree property.

Statement III: 2, 7, 10, 8, 14, 16, 20



14 is left of 10 which violates binary search tree property.

Statement IV: 4, 6, 7, 9 18, 20, 25



It doesn't violate Binary search tree property and hence it is the correct order of traversal.

Binary Search Tree Question 8

What will be post order traversal of a binary Tree T, if preorder and inorder traversals of T are given by ABCDEF and BADCFE respectively?

1. BEFDCA
2. BFDECA
3. BCFDEA
4. BDFECA

Answer (Detailed Solution Below)

Option 4 : BDFECA

Binary Search Tree Question 8 Detailed Solution

The correct answer is **option 4.**

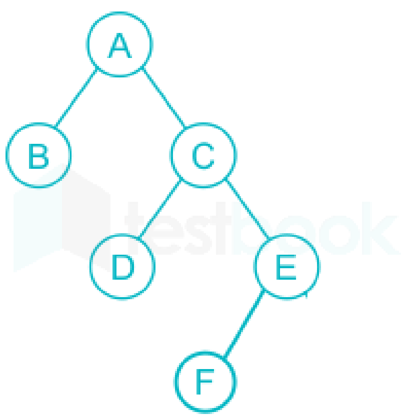
Concept:

The given data,

preorder = ABCDEF

Tree traversal			
	Inorder	**Preorder**	**Postorder**
Method Sequence	Left Sub-tree	Root	Left Sub-tree
	Root	Left Sub-tree	Right Sub-tree
	Right Sub-tree	Right Sub-tree	Root

The binary tree for the traversal is,



Post order for the above tree is,

BDFECA

Hence the correct answer is BDFECA.

The following numbers are inserted into an empty binary search tree in the given order: 10, 1, 3, 5, 15, 12, 16. What is the height of the binary search tree?

1. 3

2. 4

3. 5

4. 6

Answer (Detailed Solution Below)

Option 1 : 3

Binary Search Tree Question 9 Detailed Solution

The correct answer is **option 1**

Concept:

A binary search tree (BST) is a node-based binary tree data structure and it follows the following points

1. Left sub-tree nodes key value will exist only if lesser than the parent node key value.
2. Right sub-tree nodes key value will exist only if greater than the parent node key value.
3. Left sub-tree and Right sub-tree must be a Binary search tree.

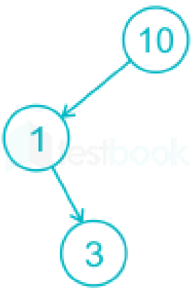
Explanation:

Step 1: First 10 comes and now that is the **Root** node.

Step 2: Now 1 came and 1 < 10 then insert Node 1 to the **Left** of Node 10.

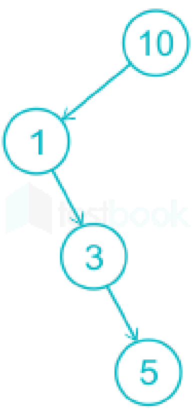
Step 3: Now 3 came and 3 < 10 go to the **Left** of

Node 10 and check 3 > 1 then insert Node 3 to the **Right** of Node 1.

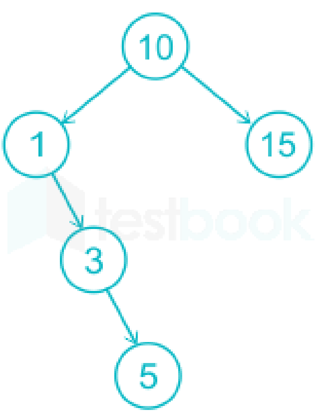


Step 4: Now 5 came and 5 < 10 go to the **Left** of

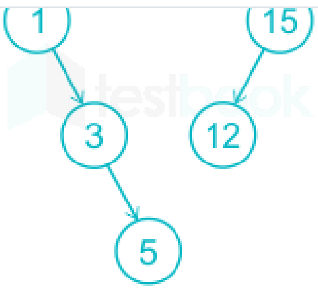
Node 10 and check 5 > 1 go to the **Right** of Node 1 then check 5 > 3 then insert Node 5 to the **Right** of Node 3.



Step 5: Now 15 came and 15 > 10 then insert Node 15 to the **Right** of Node 10.

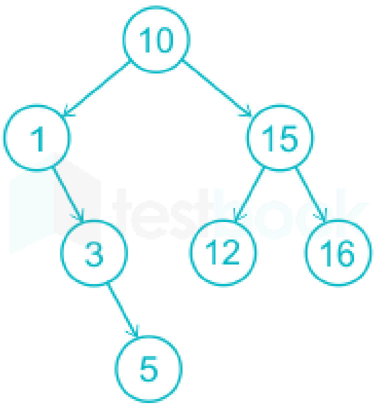


Step 6: Now 12 came and 12 > 10 go to the **Right** of Node 10 and check 15 > 12 then insert Node 12 to the **Left** of Node 15.



Step 7: Now 16 came and 16 > 10 go to the **Right** of

10 and check 16 > 15 then insert 16 to the **Right** of Node 15.



After step 7, we can count the height of the tree as 3.

Important Points

Follow the longest path in the tree and count the edges that are height.

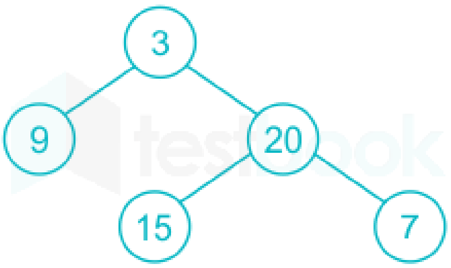
Tips To Learn:

Left sub-tree(key)<**Node**(key)<**Right sub-tree**(key)

Node(key): Parent node of **Left sub-tree** and **Right sub-tree**

Binary Search Tree Question 10

Which of the following is a height of a given binary tree?



Hint:

The height of a binary tree is equal to the largest number of edges from the root to the most distant leaf node.

1. 1
2. 2
3. 3
4. 4

Answer (Detailed Solution Below)

Option 2 : 2

Binary Search Tree Question 10 Detailed Solution

The correct answer is **option 2.**

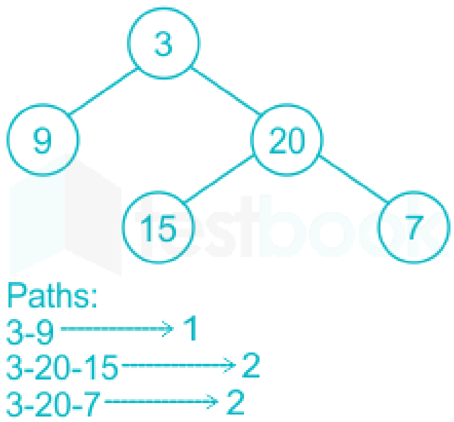
Concept:

Binary tree:

A binary tree is a tree in which no node can have more than two children. Every binary tree has parents, children, siblings, leaves, and internal nodes.

leaf node.

Explanation:



Hence the correct answer is 2.

Binary Search Tree Question 11

Suppose a binary search tree with 1000 distinct elements is also a complete binary tree. The tree is stored using the array representation of binary heap trees. Assuming that the array indices start with 0, the 3rd largest element of the tree is stored at index

Answer (Detailed Solution Below) **509**

The correct answer is 509.

Concept:

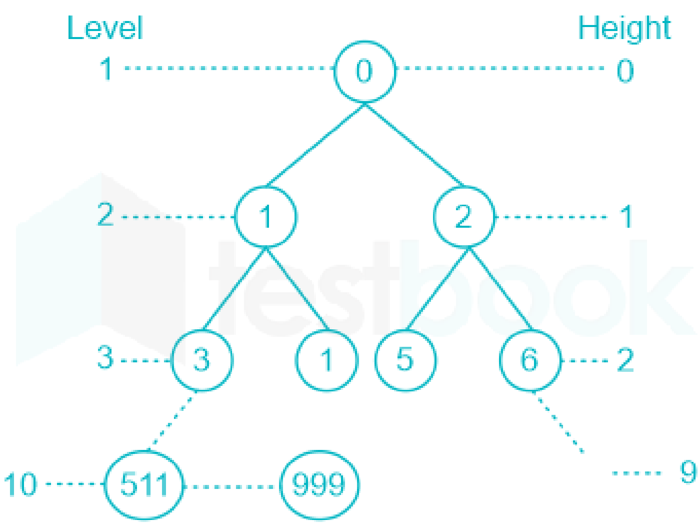
A binary search tree (BST) is a binary tree in which each node has a Comparable key (and an associated value) and the key in any node is greater than the keys in all nodes in that node's left subtree and less than the keys in all nodes in that node's right subtree.

Explanation:

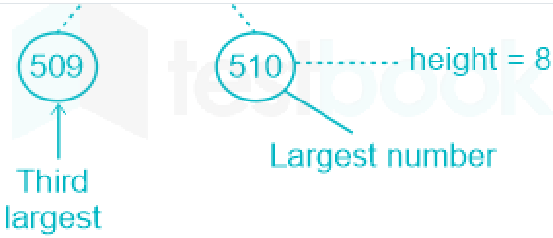
The given data,

A binary search tree with 1000 different elements has been provided. The tree is also stored using the binary heap tree's array representation. The indices of an array begin with 0.

The index number of the first node at each level may be determined using (2^h –1), where "h" is the tree's height. Also, the number of nodes at each level may be calculated using (2^n –1), where n is the number of levels.



At 10th level number of nodes = 2^{10-1} = 512.
At height 9, the index number of the first node = 2^9-1 = 511. Since the total number of nodes is 1000, we need to check the upper level. Because the rightmost number in the binary search tree is maximum.
At 9th level, no. of nodes = 2^{9-1} = 256. At height 8, index no. of first node = 2^8-1 = 255 .
The index number of the last node in 9 th level, = 255 × 2 = 510



So the third largest value is stored at = 509

Hence the correct answer is 509.

Binary Search Tree Question 12

The program written for binary search, calculates the midpoint of the span as mid : = (Low + High)/2. The program works well if the number of elements in the list is small (about 32,000) but it behaves abnormally when the number of elements is large. This can be avoided by performing the calculation as:

1. mid : = (High - Low)/2 + Low
2. mid : = (High - Low + 1)/2
3. mid : = (High - Low)/2

Answer (Detailed Solution Below)

Option 1 : mid : = (High - Low)/2 + Low

Binary Search Tree Question 12 Detailed Solution

The correct answer is **option 1.**

Key Points

- In a general scenario, binary search **mid-value** computed with, **mid=(low+high)/2**.
- However, with a **vast list** of elements, "high" would be a very **large value**. As a result, it's possible that it's **beyond** the **Integer limit**. It's known as an **integer overflow**.
- To stop this Integer overflow, the 'mid' value can also be determined using, **mid = (High - Low)/2 + Low**
- **Integer overflow** is never an issue with this form.

Explanation

mid : = (High - Low)/2 + Low

mid : = High/2 - Low/2 + low

mid : = (High + Low)/2

Alternate Method

- Option D is removed because it is the same as the incorrect option.
- Taking into account The low index is 10, while the high index is 15.
- Option B returns a mid-index of 3 that is not even in the sub-array index.
- Choice C returns a mid-index of 2 that isn't even in the sub-array index.
- Option A is the best solution.

Hence the correct answer is *mid : = (High - Low)/2 + Low* .

Binary Search Tree Question 13

A binary search tree is constructed by inserting the following numbers in order:

60, 25, 72, 15, 30, 68, 101, 13, 18, 47, 70, 34

The number of nodes is the left subtree is

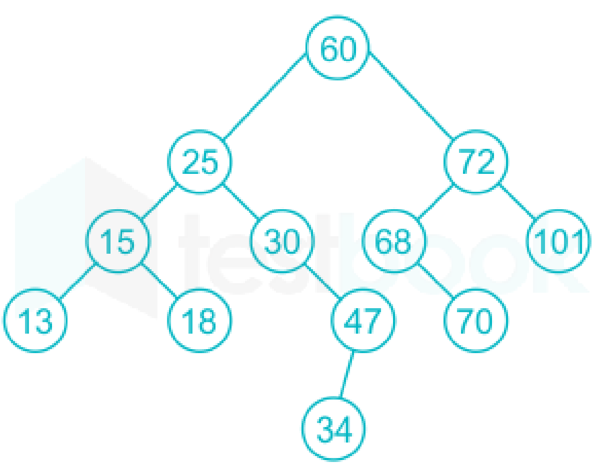
1. 7
2. 6
3. 3
4. 5

Answer (Detailed Solution Below)

Option 1 : 7

Binary Search Tree Question 13 Detailed Solution

Binary Search Tree (BST):



Tips and Tricks:

Sorted the nodes in BST: 13 15 18 25 30 34 47 **60** 68 70 72 101

Count the nodes to the left of the root node (60).

Binary Search Tree Question 14

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 reversal ordering on natural numbers i.e. 9 is assumed to be smallest and 0 is assumed to be largest. The in-order traversal of the resultant binary search tree is

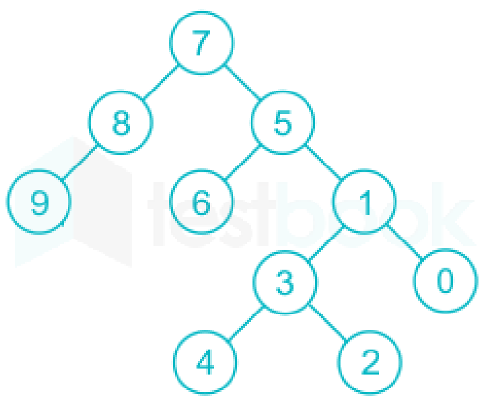
1. 9, 8, 6, 4, 2, 3, 0, 1, 5, 7
2. 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
3. 0, 2, 4, 3, 1, 6, 5, 9, 8, 7
4. 9, 8, 7, 6, 5, 4, 3, 2, 1, 0

Answer (Detailed Solution Below)

Binary Search Tree Question 14 Detailed Solution

Insert 7, 5, 1, 8, 3, 6, 0, 9, 4, 2 in empty binary search tree by using reverser ordering

Binary search tree:



Inorder traversal = 9, 8, 7, 6, 5, 4, 3, 2, 1, 0

Tips and Trick:

In-order of the binary tree is always sorted in ascending order but binary search tree uses the reversal ordering on natural numbers

Therefore it is sorted in descending order:

Binary Search Tree Question 15

1. θ (log n) for both insertion and deletion
2. θ (n) for both insertion and deletion
3. θ (n) for insertion and θ (log n) for deletion
4. θ (log n) for insertion and θ (n) for deletion

Answer (Detailed Solution Below)

Option 2 : θ (n) for both insertion and deletion

Binary Search Tree Question 15 Detailed Solution

Concepts:

Minimum height of the tree is when all the levels of the binary search tree (BST) are completely filled.

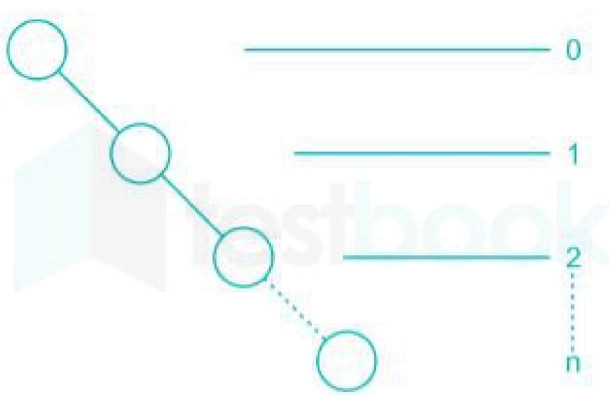
Maximum height of the BST is the worst case when nodes are in skewed manner.

Formula:

Minimum height of the BST with n nodes is $\lceil \log_2 (n + 1) \rceil - 1$

The maximum height of the BST with n nodes is n - 1.

BST with a maximum height:



Insertion:

Deletion:

Traverser the BST to the maximum height

Worst-case time complexity of $= \theta\ (n - 1) \equiv \theta\ (n)$