1. Definition of AVL Tree and Its Advantages

An AVL tree is a self-balancing binary search tree in which the height difference between the left and right subtrees of any node is limited. This height difference is called the balance factor. The balance factor can be -1, 0, or +1. AVL trees automatically maintain balance after insertion and deletion operations. The main advantage of an AVL tree is that it guarantees faster searching because the tree remains balanced. It also provides efficient insertion and deletion operations and ensures better performance compared to unbalanced binary search trees.

2. Difference Between AVL Tree and B-Tree

An AVL tree is a binary search tree where each node can have at most two children and the tree remains balanced using rotations. A B-tree is a multi-level tree where each node can have multiple keys and children. AVL trees are mainly used in memory-based applications, while B-trees are widely used in disk-based storage systems. AVL trees focus on maintaining height balance, whereas B-trees focus on minimizing disk access by storing more keys in a single node.

3. Deletion of an Item from B-Tree

Deletion in a B-tree is a complex process because the tree must maintain its properties after deletion. When an item is deleted, the tree ensures that each node has the required minimum number of keys. If a node has fewer keys than required, keys are borrowed from sibling nodes or nodes are merged. This process continues until the tree structure becomes valid. Deletion maintains balance and ensures efficient searching.

4. Structure and Operations of B-Tree

A B-tree is a balanced tree data structure where each node can contain multiple keys and children. All leaf nodes are at the same level. The structure ensures that the tree remains balanced. Operations of a B-tree include searching, insertion, and deletion. Searching is done by comparing keys within nodes. Insertion may cause node splitting, while deletion may cause merging or redistribution of keys.

5. Insertion of an Item in B-Tree

Insertion in a B-tree starts from the root node. The correct position for the new key is found. If the node has space, the key is inserted. If the node is full, it is split into two nodes and the middle key is moved to the parent node. This process may continue up to the root. Insertion maintains the balanced structure of the B-tree.

6. Difference Between Left Heavy Tree and Right Heavy Tree

A left heavy tree is a tree in which the height of the left subtree is greater than the height of the right subtree. A right heavy tree is a tree in which the height of the right subtree is greater than the height of the left subtree. These imbalances affect tree performance and are corrected using rotations in AVL trees.

```
Left Heavy Tree:          Right Heavy Tree:

      A                         A
     /                           \
    B                             C
   /                               \
  D                                 E
```

7. AVL Tree Rotations

AVL tree rotations are used to maintain balance after insertion or deletion. There are four types of rotations. Single right rotation is used when the tree becomes left heavy. Single left rotation is used when the tree becomes right heavy. Double rotations include left-right rotation and right-left rotation. These rotations adjust the structure of the tree and restore balance while maintaining binary search tree properties.

```
1. Left Rotation
2. Right Rotation
3. Left-Right Rotation
4. Right-Left Rotation
```