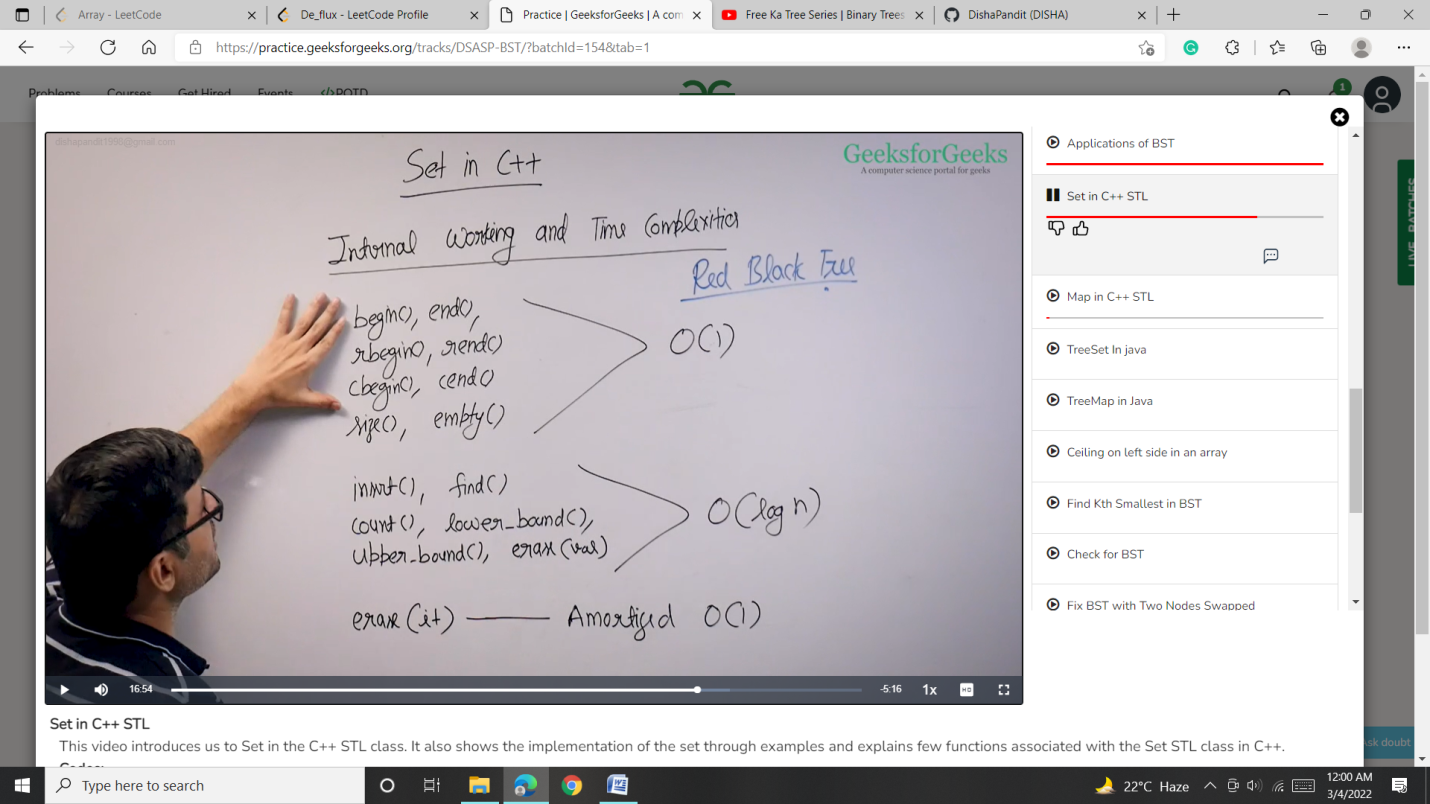
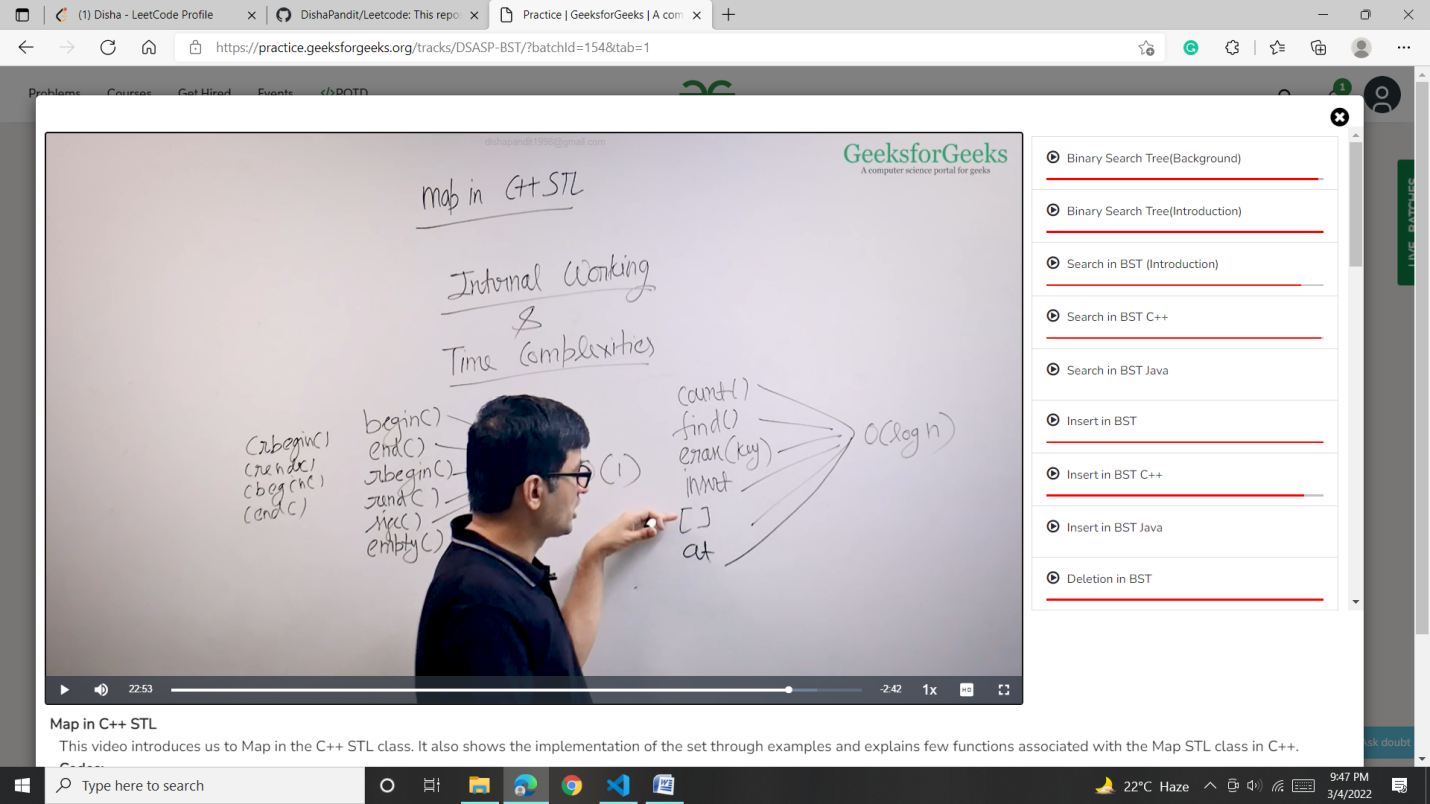
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Array(Unsorted) | Array(Sorted) | Linked List | BST | Hash Table |
| Search | O(n) | O(logn) | O(n) | O(logn) |  |
| Insert | O(1) | O(n) | O(1),  O(n) for sorted | O(logn) | O(1) |
| Delete | O(n) | O(n) | O(n) | O(logn) | O(1) |
| Find closest | O(n) |  | O(n) | O(logn) | O(n) |
| Sorted Traversal | O(nlogn) |  | O(nlogn),  O(n) in case of sorted) | O(n) | O(nlogn) |

* We get O(logn) in BST only when it is balanced. For skewed BST, TC is O(n). In general, we say TC is O(height) for BST.
* BST – For every node, keys on its left are smaller and keys on its right are greater.
* In CPP, BST is implemented using map, set, multimap, multiset.
* There is a specific order to insert and delete element in BST which is not there in binary tree.
* Inorder Traversal of BST is a sorted sequence.
* A new key is always inserted at the leaf.

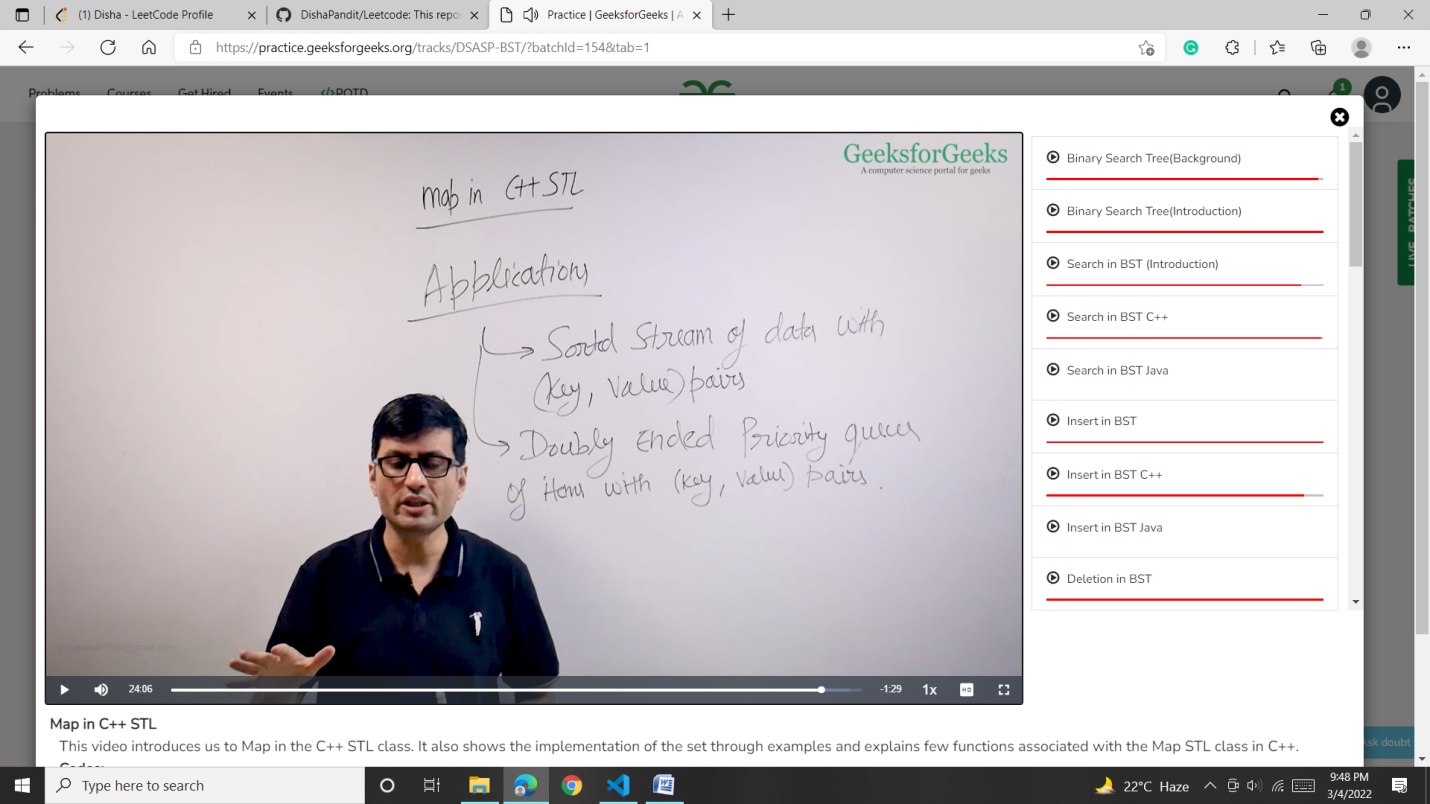
Set : set is based on Red black tree. When we traverse a BST in inorder fashion, we get sorted order.

That is why when we traverse a set we get sorted order





Application of set and map are same



Min element is at .begin()

Max element is at .end()-1