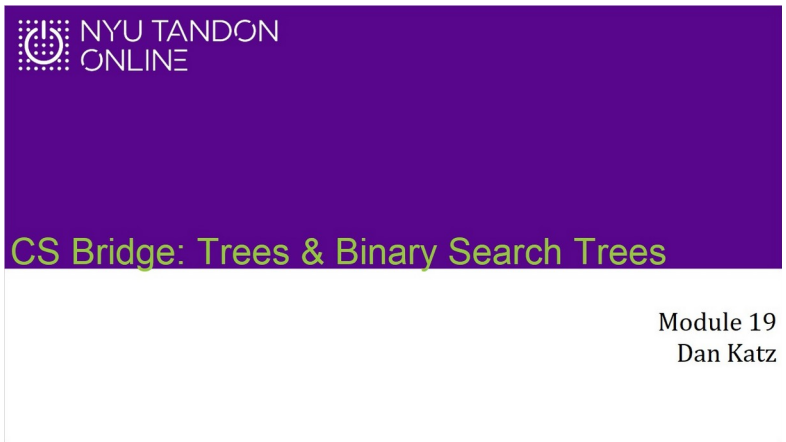
# CS Bridge Module 19 Trees & Binary Search Trees

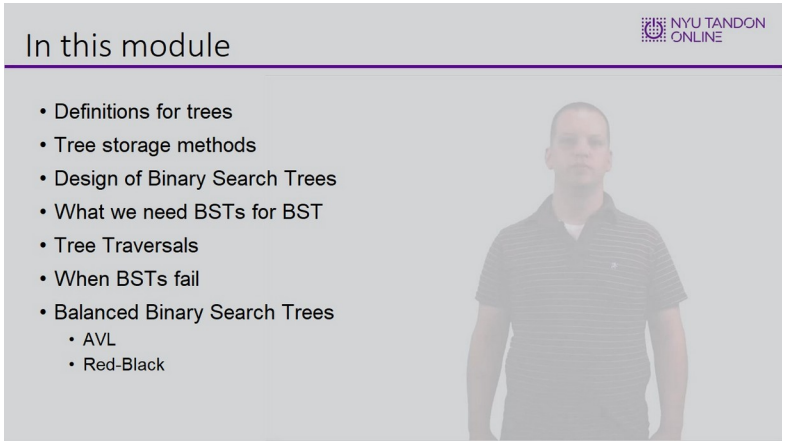
## 1. Trees and BST

### *1.1 CS Bridge: Trees & Binary Search Trees*



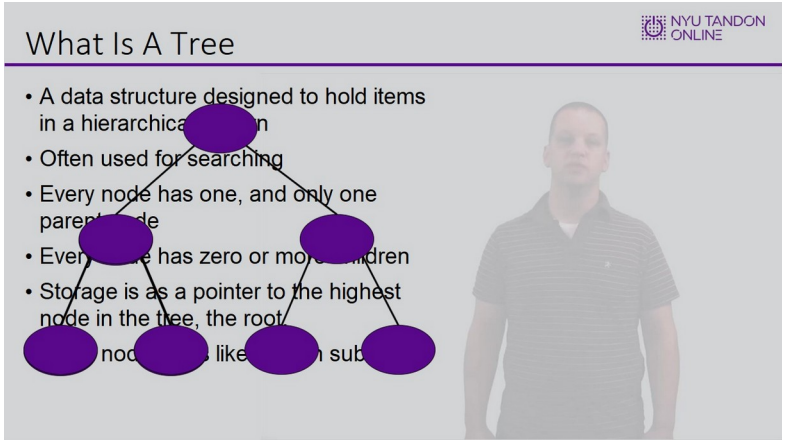

**Notes:**

### *1.2 In this module*

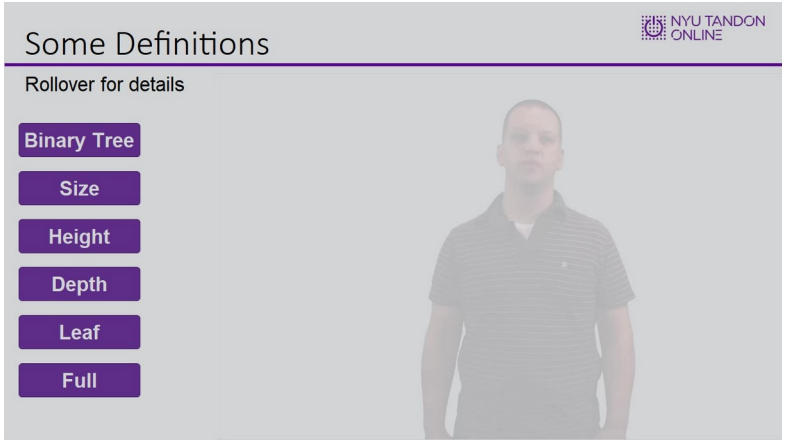



**Notes:**

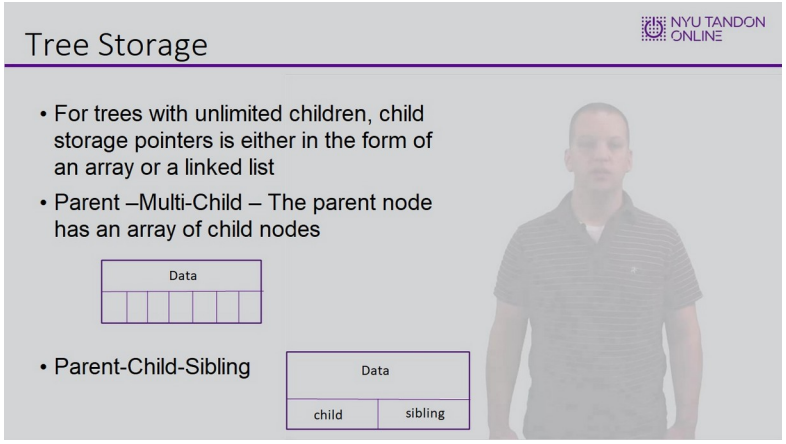
### *1.3 What is a tree*



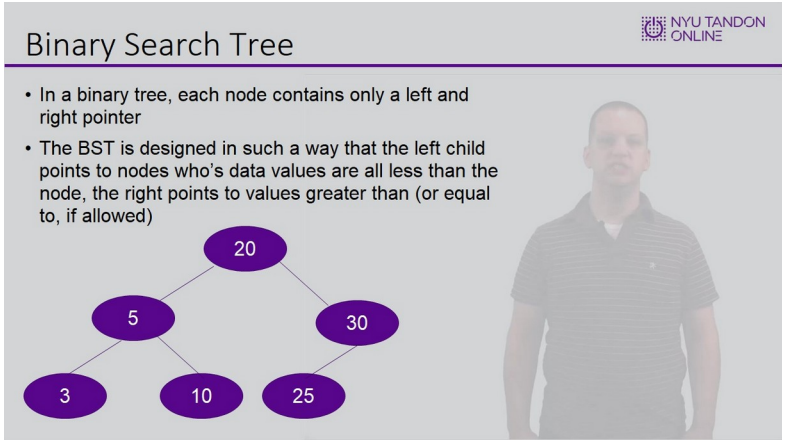

### *1.4 Some definitions*



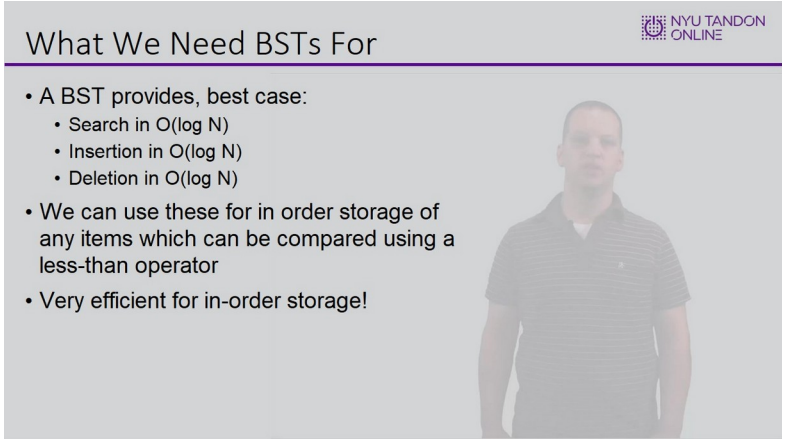
## *1.5 Tree Storage*




## *1.6 Binary Search Tree*



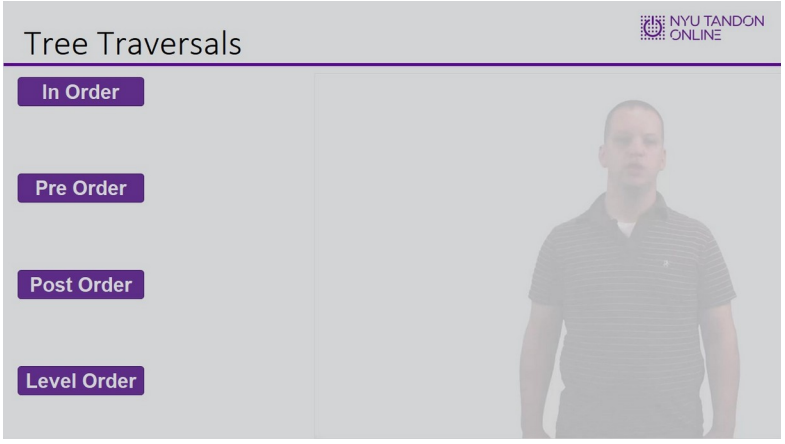

## *1.7 What we need BSTs for*



## *1.8 BST Node code*

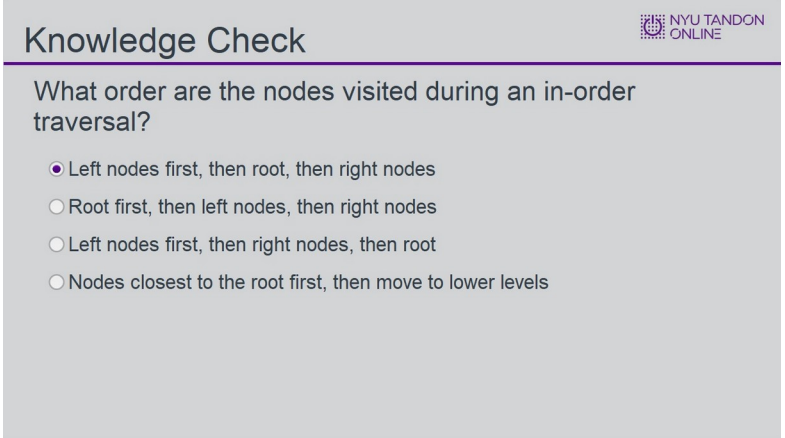

## *1.9 Recursion in trees*


## *1.10 Tree Traversals*



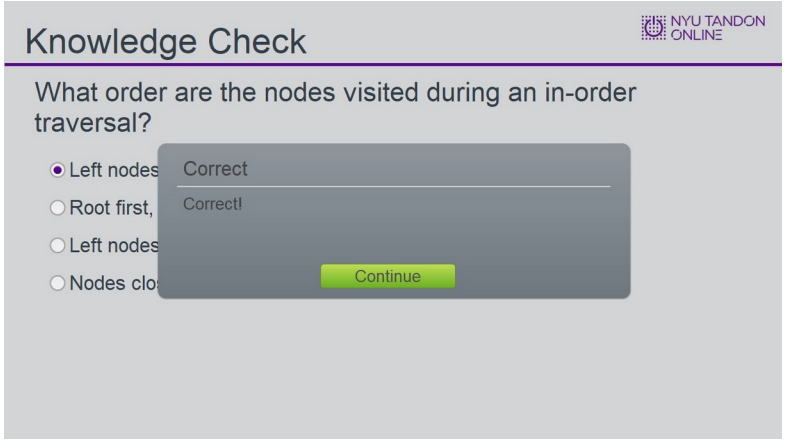
### *1.11 Knowledge Check*

*(Multiple Choice, 10 points, 1 attempt permitted)*

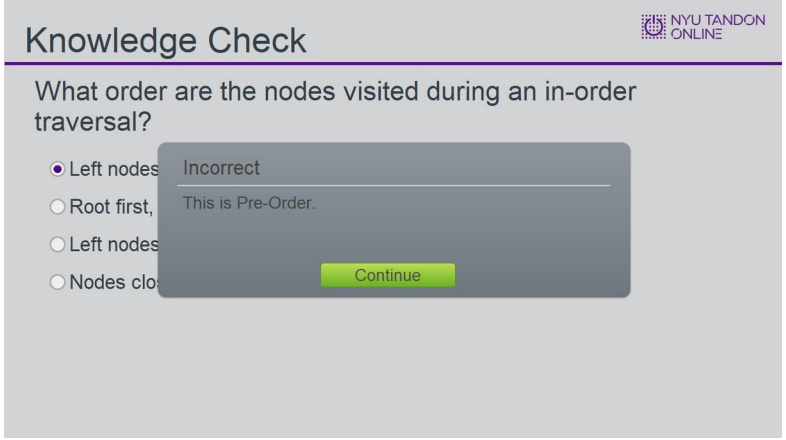



| Correct | Choice | Feedback |
|---|---|---|
| X | Left nodes first, then root, then right nodes | Correct! |
| | Root first, then left nodes, then right nodes | This is Pre-Order. |
| | Left nodes first, then right nodes, then root | This is Post Order. |
| | Nodes closest to the root first, then move to lower levels | This is Level Order. |

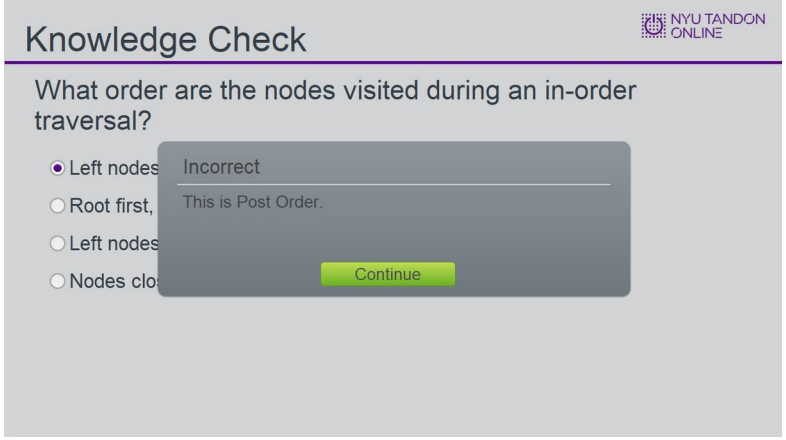
**Incorrect (Slide Layer)**



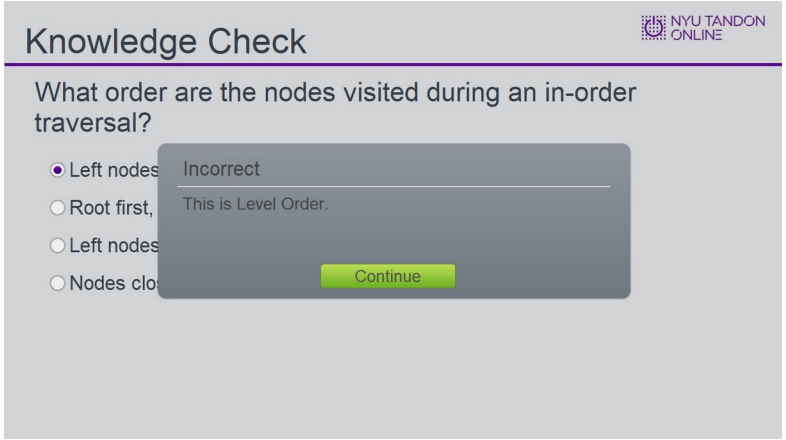

**Incorrect (Slide Layer)**



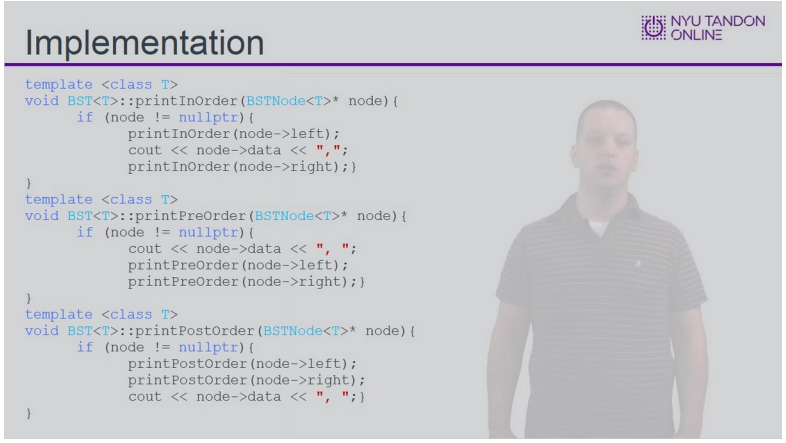

**Incorrect (Slide Layer)**



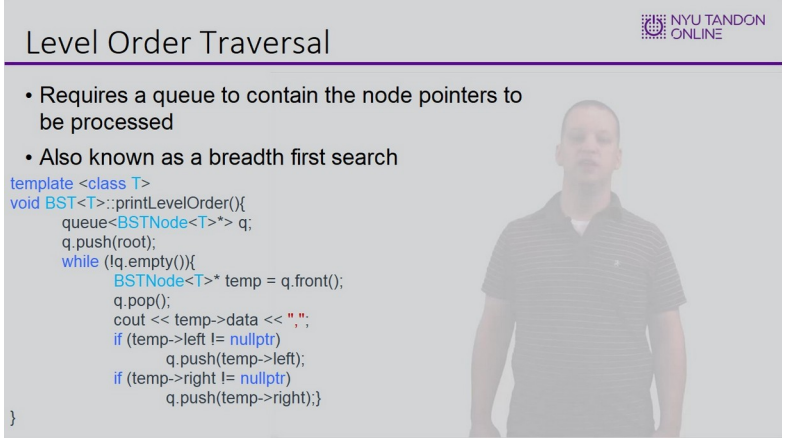
**Incorrect (Slide Layer)**



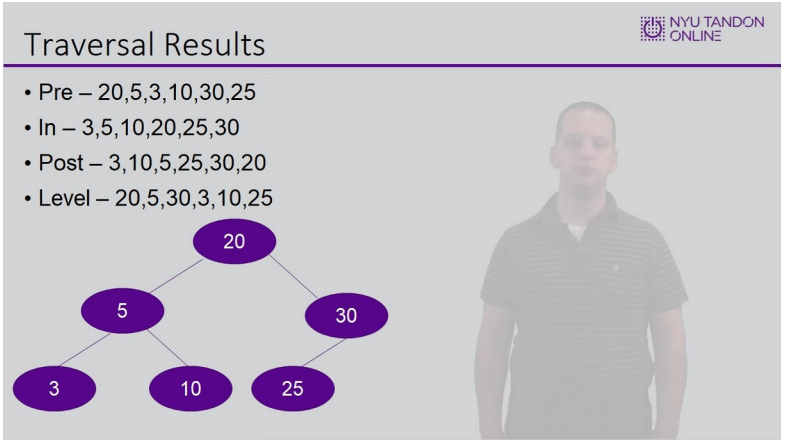

## *1.12 Implementation*



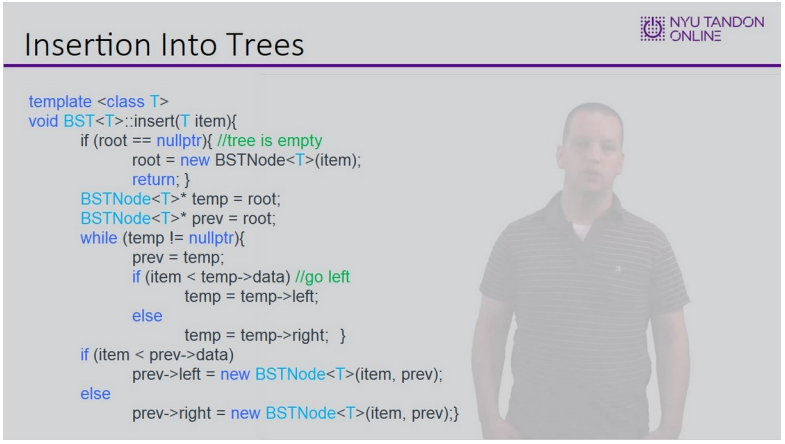

## *1.13 Level order traversal*



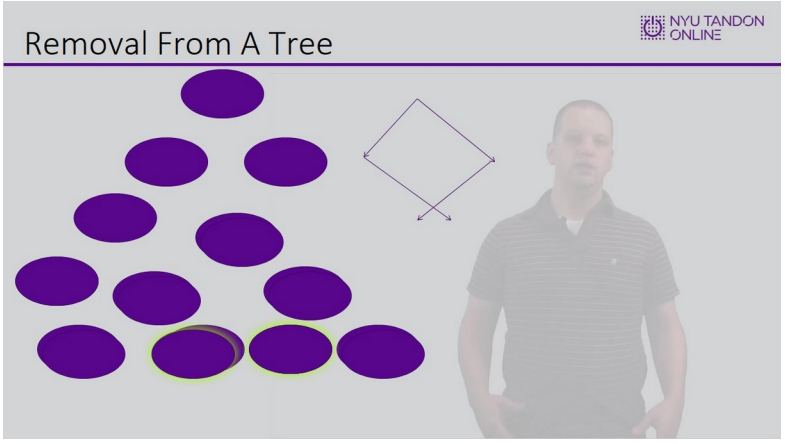
### *1.14 Traversal results*



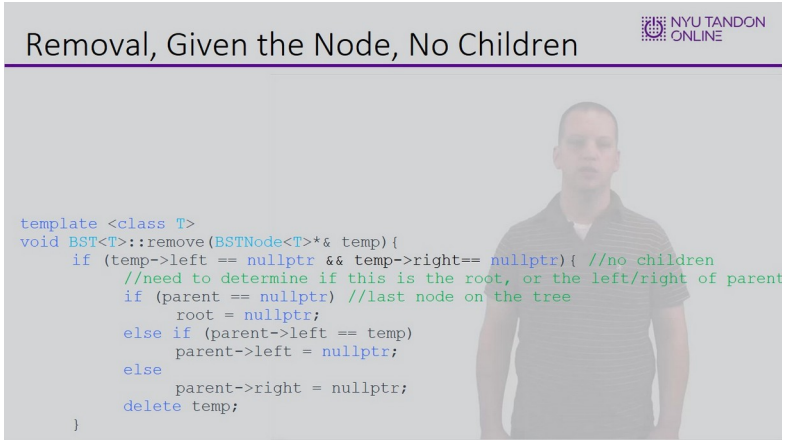

### *1.15 Insertion into trees*



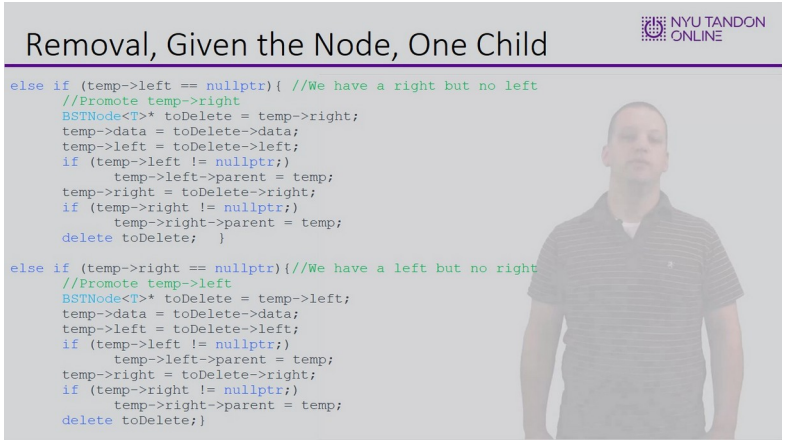

### *1.16 Removal from a tree*



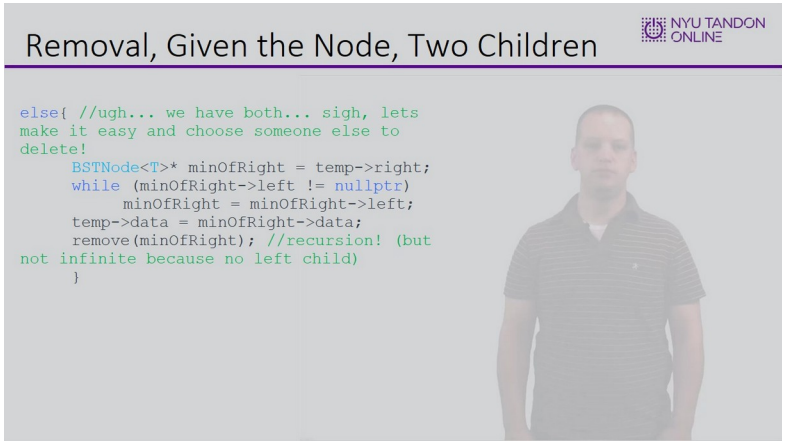
## *1.17 Removal, given the node, no children*



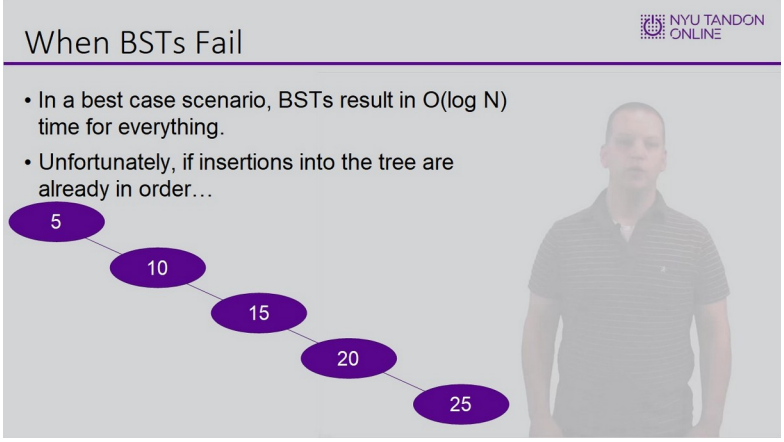

## *1.18 Removal, given the node, one child*



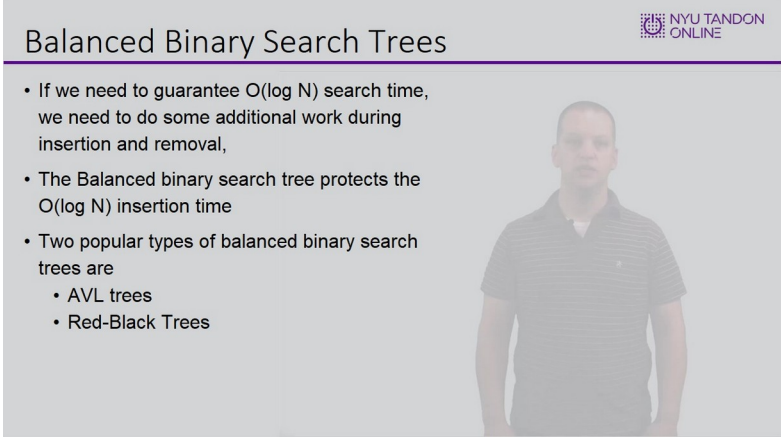

## *1.19 Removal, Given the node, two children*



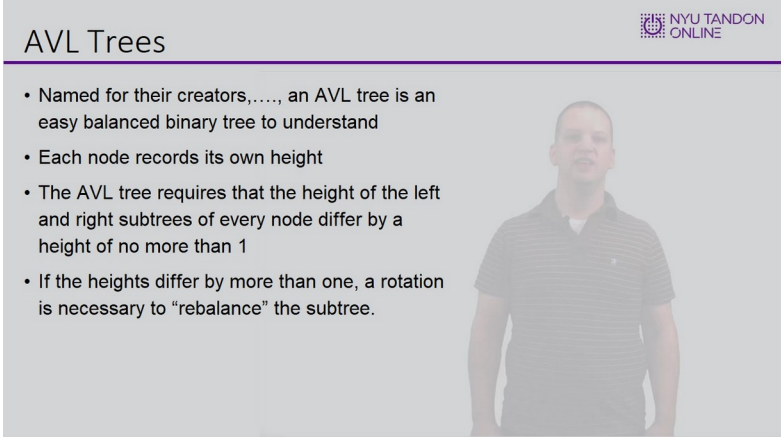
### *1.20 When BSTs fail*



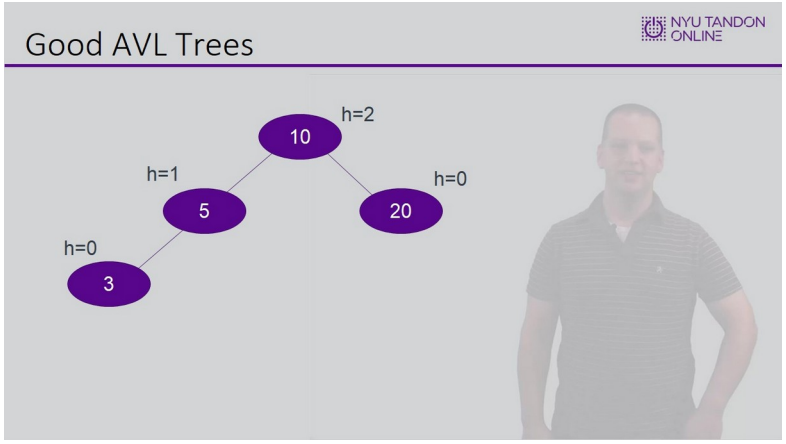

### *1.21 Balanced Binary Search Trees*



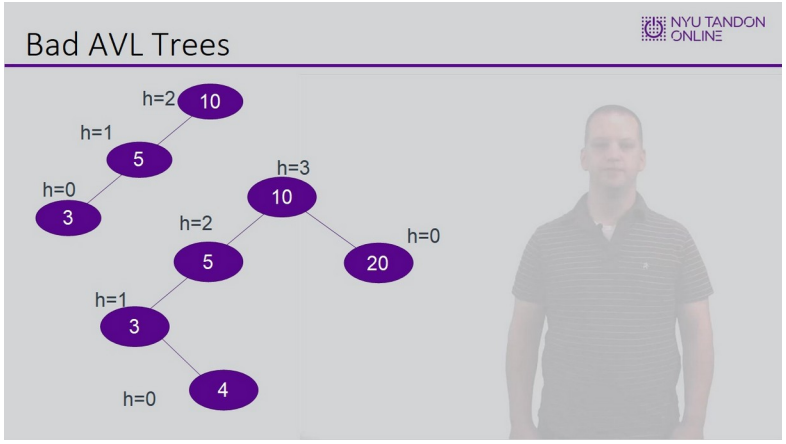

### *1.22 AVL Trees*



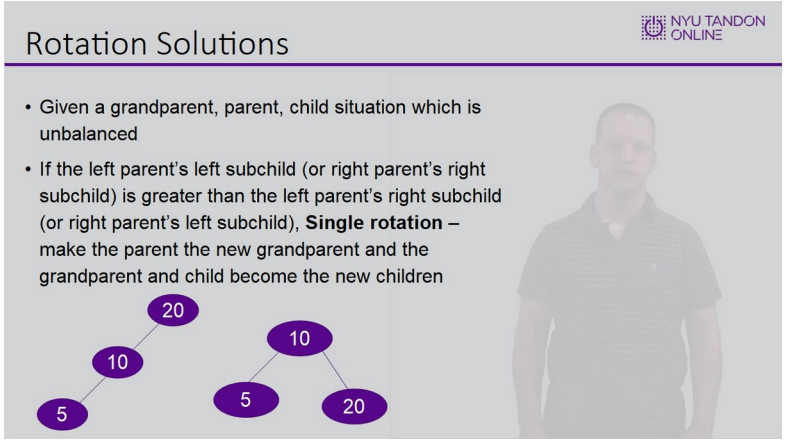
### *1.23 Good AVL trees*



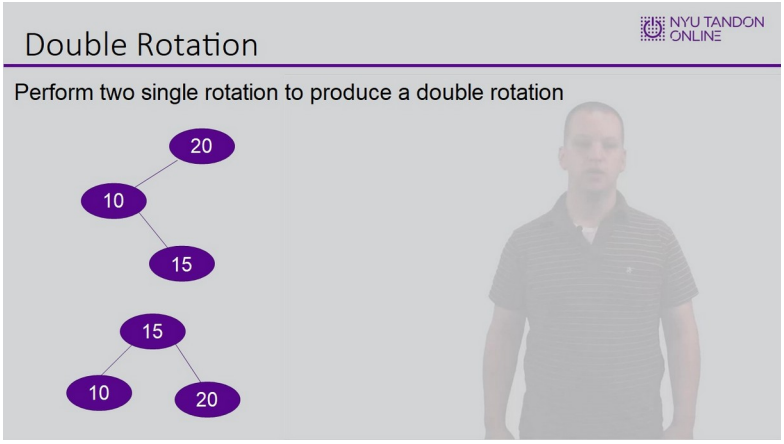

### *1.24 Bad AVL trees*




### *1.25 Rotation solutions*

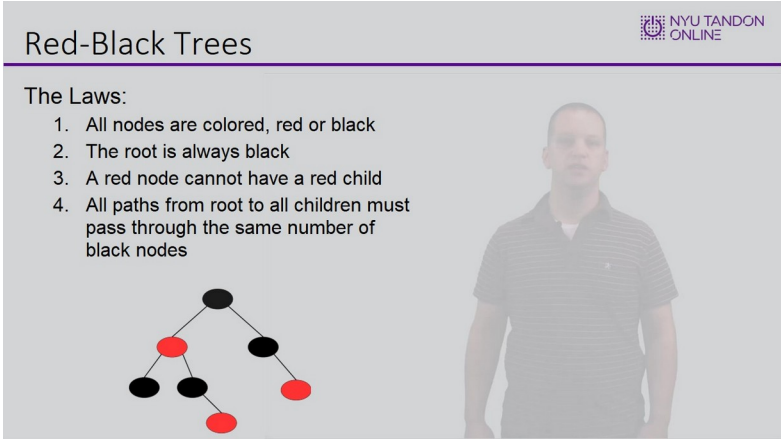


## *1.26 Double Rotation*

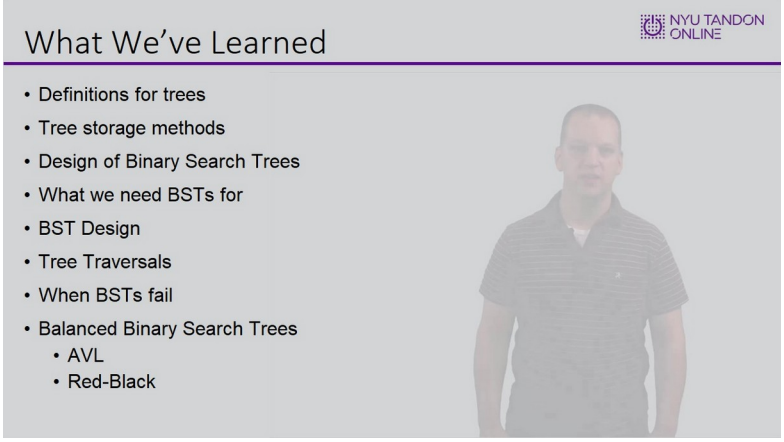



**Notes:**

## *1.27 Red-Black Trees*



### *1.28 In this module, we learned*




### *1.29 End of Module*