upGrad

Full-Stack Software Development

upGrad

Lecture On: BST

Instructor: Arkoprovo Dey

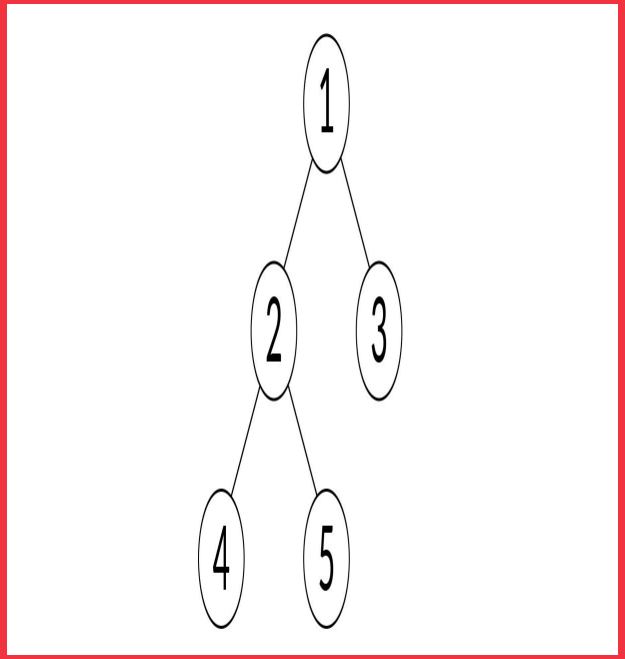
In Last Class, we covered….

- Trees

Homework Discussion

You are given a binary tree. You need to print the maximum height of the binary tree. If the tree is NULL (empty tree), print the height of the tree as 0.

Output: 3
Here, in the given tree, the maximum height of the binary tree is 3.



Today's Agenda

1 Binary Search Tree

Binary Search Trees

upGrad

Binary Search Trees

upGrad

Previously, you have learnt how Binary Trees work and how one traverses them. Today, we will explore a special kind of Binary Trees.

Do you remember how we had implemented a Binary Search? It required us to have a linear data structure that was sorted. Now, let us revisit how we had thought about searching for words in a dictionary after being inspired by Binary Search.

Can you think of a few problems that we might run into while using a Binary Search?

Binary Search Trees

upGrad

What if now you need to include a new word (or maybe a few hundred words) in the English dictionary? How would you go about inserting this new data into the already sorted array?

What would its time complexity be?

Binary Search Trees

upGrad

What if you were required to remove a few words now? How would you go about doing that?

What would be the time complexity of such operations?

Binary Search Trees

upGrad

A Binary Search Tree provides us with a solution to some of these problems. It is a special kind of Binary Tree that helps one traverse the tree to search for data.

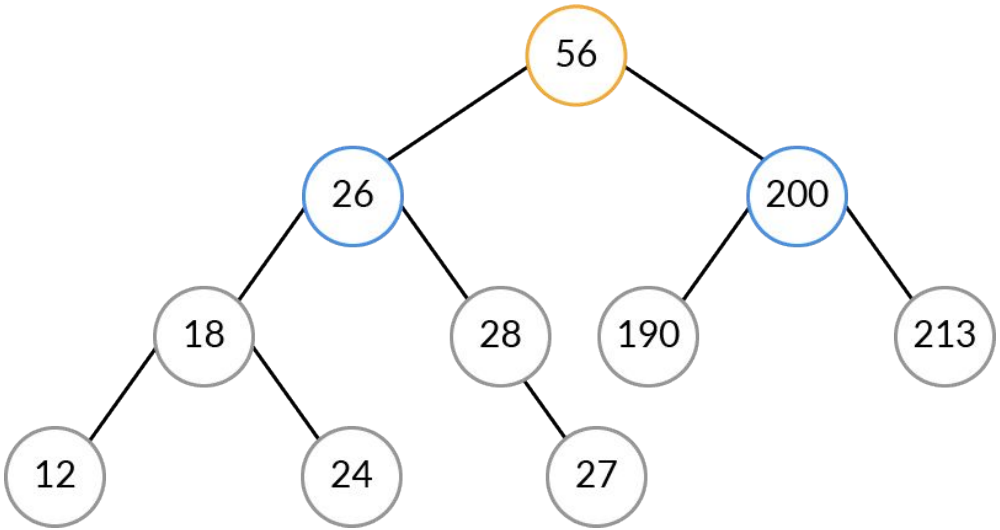
In order to do this, it must adhere to these simple properties. A Binary Search Tree is a **binary** tree where, for all the nodes in the tree,

- All the data elements in the left subtree of the node are less than or equal to the node in consideration if the data elements were to be sorted.
- All the data elements in the right subtree of the node are greater than the node in consideration if the data elements were to be sorted.

Binary Search Trees

upGrad

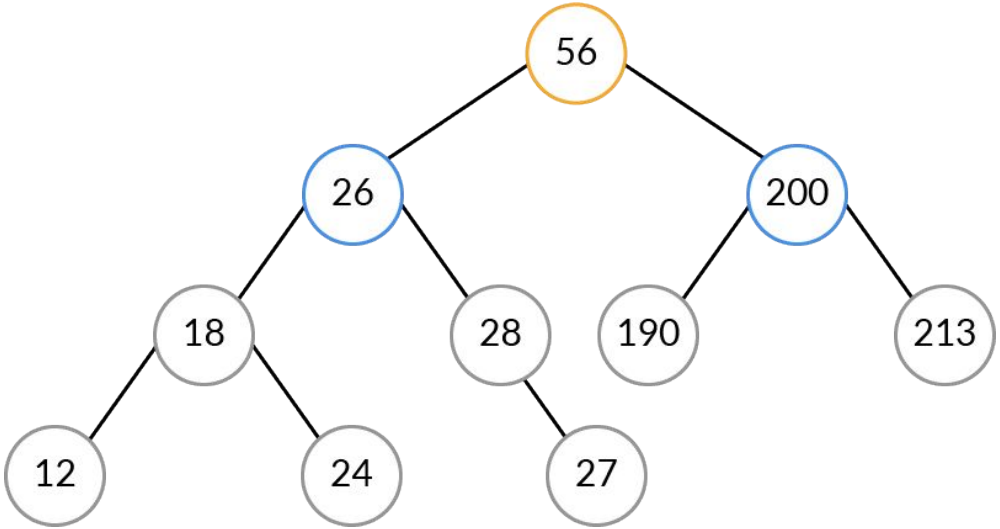
Whenever we perform an operation on a Binary Search Tree, be it creation, insertion, search or deletion, we must ensure that we do not violate these properties.



Binary Search Trees

upGrad

What would be the in-order traversal of this Binary Search Tree? Can you notice anything about the in-order traversal?



Binary Search Trees

upGrad

If the Binary Search Tree is more or less *complete*, what would be the time complexity in searching for an element in that tree?

Think whether you can take inspiration from how we did it in a Binary Search.

Binary Search Trees

upGrad

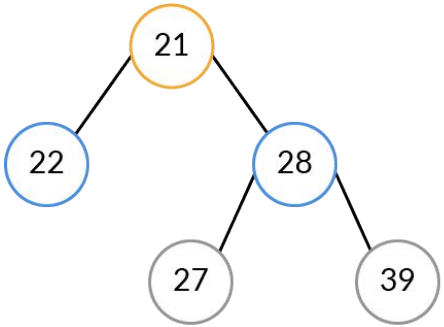
What would the situation be if the tree was degenerated? What would the search time complexity be?

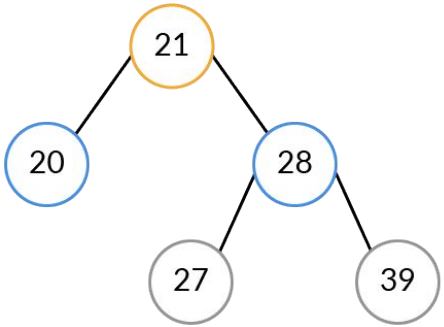
Can you think how we might end up with such a Binary Search Tree?

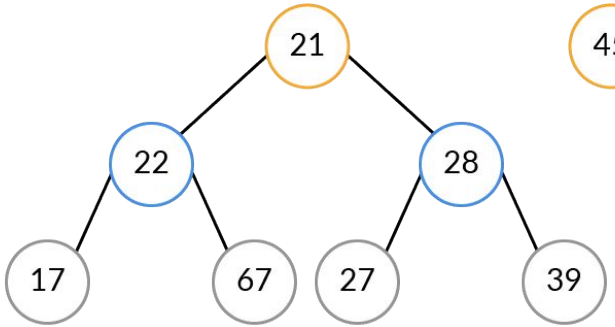
Binary Search Trees

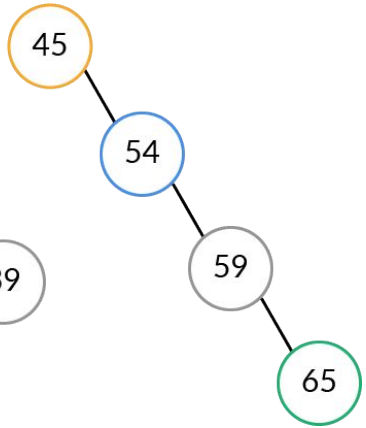
upGrad

Are these trees Binary Search Trees?





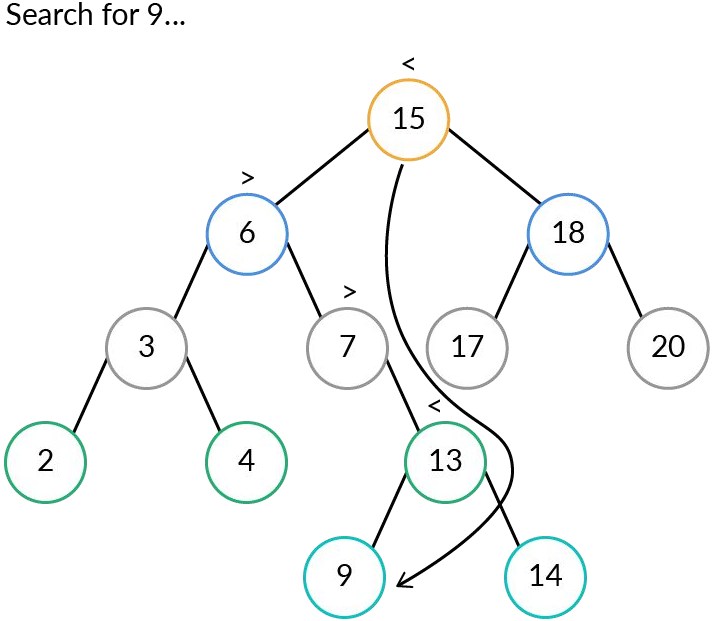




Binary Search Trees

upGrad

Let us now learn how we would go about searching for an element in a Binary Search Tree.



Binary Search Trees

upGrad

Let us now learn how we would go about searching for an element in a Binary Search Tree.

```
begin
  if (node > data == key OR node == NULL) then
      return node;
  else
      if (key < node > data) then
            return binTreeSearch(node>left, key);
      else
            return binTreeSearch(node>right, key);
      end if
  end if
end
```

But searching an element is something we have already done in previous lessons. The main beauty of Binary Search Trees is the simplicity of its data insertion (and by extension, creation) and data deletion.

We already know that inserting data elements into an already sorted list of ‘n’ elements is going to be around O(n). Now, let us see how Binary Search Trees help us achieve something better.

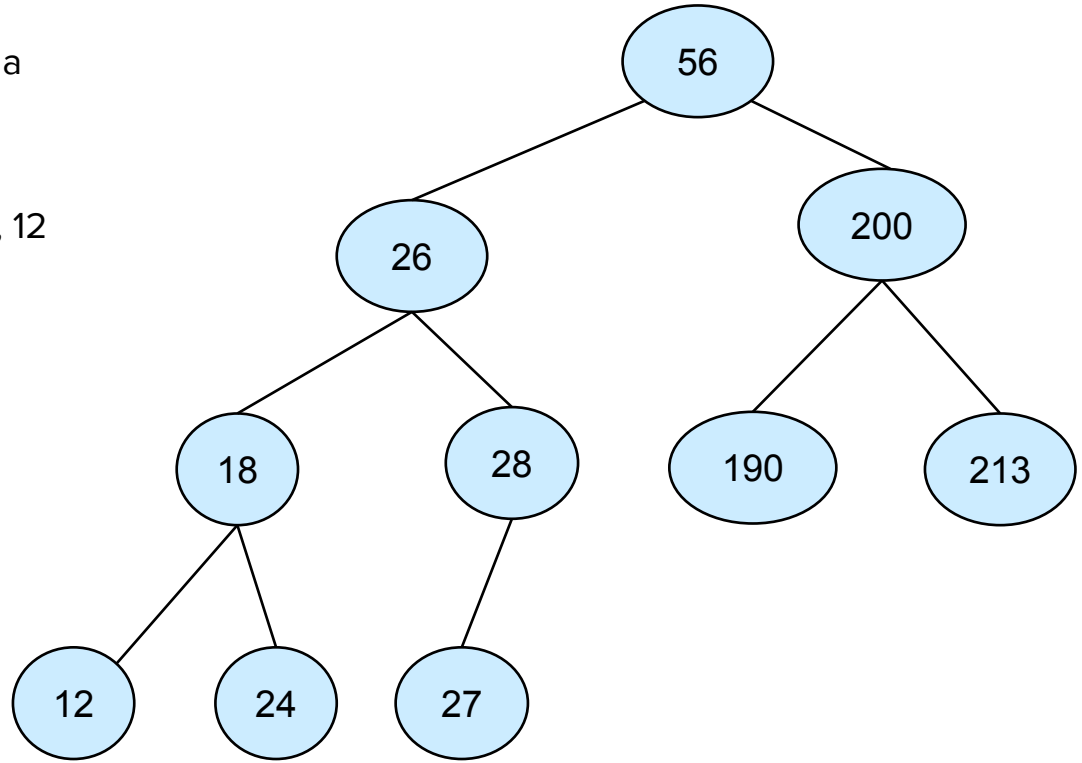
First, we shall learn how we can create a Binary Search Tree and insert new data elements into it.

Binary Search Trees

upGrad

Suppose we are required to create a Binary Search Tree out of the elements:

56, 26, 28, 200, 27, 18, 213, 24, 190, 12



Binary Search Trees

upGrad

What is the time complexity of inserting a new node into a Binary Search Tree?

Binary Search Trees

upGrad

Given a Binary Search Tree, how would you find the minimum element in it? What would be the worst-case time complexity that you can achieve?

Similarly, how would you find out the maximum element in the tree? What would its worst-case time complexity be?

Homework

You will be given a binary search tree and a target sum, you have to find whether there are any two nodes in the BST whose sum is equal to the given target sum.

Tasks to complete after the session

Homework Questions
MCQs
Coding Questions

In the next class...

- Binary Search Tree

upGrad

#RahoAmbitious

Thank You!