In our previous video, we looked at an implementation of binary tree traversal in C#, which included methods for inserting items into the tree and traversing the tree in various ways. Now we want to modify the code to add a Contains method that allows us to search for specific items in the tree.

We start by adding a new public method to the BinTree class called Contains. This method takes an integer item as a parameter and returns true if the item is present in the tree and false otherwise.

Inside the Contains method, we call a private version of the method called Contains that takes a node and an item as parameters. This private method searches the tree for the given item by recursively searching the left or right subtree, depending on whether the item is smaller or larger than the data in the current node of the tree.

The private Contains method checks if the current node is null, in which case it returns false. If the current node's value is equal to the item we're searching for, it returns true. If the item is less than the current node's value, it searches the left subtree. Otherwise, it searches the right subtree.

We then add at least two tests to the Main method to show that the Contains method works. We search for an item that is present in the tree, and another item that is not present in the tree. We use if statements to display a suitable message to the console based on the results of the Contains method.

And that's how we add a Contains method to our binary tree implementation in C#!