|  |  |
| --- | --- |
| **Search in C++** | |
| #include <iostream>  using namespace std;  // Define Node structure for BST  struct Node {  int key;  Node \*left, \*right;    Node(int item) {  key = item;  left = nullptr;  right = nullptr;  }  };  // Function to search for a node in BST  bool searchInBST(Node\* root, int k) {  if (root == nullptr) {  return false;  }  if (root->key == k) {  return true;  }  if (k < root->key) {  return searchInBST(root->left, k);  }  if (k > root->key) {  return searchInBST(root->right, k);  }  return false;  }  int main() {  // Create the BST  Node\* root = new Node(6);  root->left = new Node(3);  root->right = new Node(8);  root->right->left = new Node(7);  root->right->right = new Node(9);  // Search for nodes from 0 to 9  for (int i = 0; i < 10; i++) {  cout << i << " is Present? " << (searchInBST(root, i) ? "Yes" : "No") << endl;  }  return 0;  } | BST Structure: 6  / \  3 8  / \  7 9 🔍 Dry Run Table (Step-by-step trace of function calls):  | **Value k** | **Function Calls** | **Found?** | | --- | --- | --- | | 0 | 6 → 3 → nullptr | No | | 1 | 6 → 3 → nullptr | No | | 2 | 6 → 3 → nullptr | No | | 3 | 6 → 3 | ✅ Yes | | 4 | 6 → 3 → nullptr | No | | 5 | 6 → 3 → nullptr | No | | 6 | 6 | ✅ Yes | | 7 | 6 → 8 → 7 | ✅ Yes | | 8 | 6 → 8 | ✅ Yes | | 9 | 6 → 8 → 9 | ✅ Yes |  🖨 Output: 0 is Present? No  1 is Present? No  2 is Present? No  3 is Present? Yes  4 is Present? No  5 is Present? No  6 is Present? Yes  7 is Present? Yes  8 is Present? Yes  9 is Present? Yes |
| 0 is Present? No  1 is Present? No  2 is Present? No  3 is Present? Yes  4 is Present? No  5 is Present? No  6 is Present? Yes  7 is Present? Yes  8 is Present? Yes  9 is Present? Yes | |