



# University Of Asia Pacific

## Department of CSE

**Course Code : CSE 208**

**Course Title : Data Structures and Algorithms II Lab**

**No Of Assignment : 02**

**Date of Performance : 19.09.2024**

**Date of Submission : 26.09.2024**

### Submitted By:

**Name : Nusrat Ahmmed Ekra**

**Student Id : 22201251**

**Section : E2**

**Semester : 2<sup>nd</sup> Year 2<sup>nd</sup> Semester**

### Submitted To:

**Suri Dipannita Sayeed**

**Lecturer,**

**Department of CSE, UAP**

## Problem no: 02: Implement insertion and deletion in a balanced binary search tree.

### Code:

```
BST INSERT_DELETE_SEARCH.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global> search(struct node* root, int key, int& index, string path = "", int currentIndex = 0): node
Start here x BST INSERT_DELETE_SEARCH.cpp x
1 #include <iostream>
2 using namespace std;
3
4 struct node {
5     int key;
6     struct node* left;
7     struct node* right;
8 };
9
10 struct node* newNode(int element) {
11     struct node* temp = new node();
12     temp->key = element;
13     temp->left = temp->right = NULL;
14     return temp;
15 }
16
17 struct node* findMin(struct node* root) {
18     while (root && root->left != NULL) {
19         root = root->left;
20     }
21     return root;
22 }
23
24 struct node* search(struct node* root, int key, int& index, string path = "", int currentIndex = 0)
25
E:\BST INSERT_DELETE_SEARCH.cpp C/C++ Windows (CR+LF) WINDOWS-1252 Line 35, Col 1, Pos 766 Insert Read/Write default 1:47 AM 9/24/2024
Rain showers
BST INSERT_DELETE_SEARCH.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global> search(struct node* root, int key, int& index, string path = "", int currentIndex = 0): node
Start here x BST INSERT_DELETE_SEARCH.cpp x
22 }
23
24 struct node* search(struct node* root, int key, int& index, string path = "", int currentIndex = 0)
25 {
26     if (root == NULL) {
27         return NULL;
28     }
29     if (root->key == key) {
30         index = currentIndex;
31         cout << "Found at index: " << index << " (Path: " << path << ") << endl;
32         return root;
33     }
34
35     struct node* leftResult = search(root->left, key, index, path + "left -> ", currentIndex + 1);
36     if (leftResult != NULL) {
37         return leftResult;
38     }
39
40     return search(root->right, key, index, path + "right -> ", currentIndex + 1);
41 }
42
43
44
45 struct node* insertNode(struct node* Node, int a) {
```

```
BST INSERT_DELETE_SEARCH.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global> search(struct node* root, int key, int& index, string path = "", int currentIndex = 0): node
Start here x BST INSERT_DELETE_SEARCH.cpp x
43 }
44
45 struct node* insertNode(struct node* Node, int a) {
46     if (Node == NULL) {
47         return newNode(a);
48     }
49     if (a < Node->key) {
50         Node->left = insertNode(Node->left, a);
51     } else if (a > Node->key) {
52         Node->right = insertNode(Node->right, a);
53     }
54     return Node;
55 }
56
57
58 void preOrder(struct node* root) {
59     if (root != NULL) {
60         cout << " " << root->key << " ";
61         preOrder(root->left);
62         preOrder(root->right);
63     }
64 }
65
66 struct node* deleteNode(struct node* root, int t) {
```

```
E:\BST INSERT_DELETE_SEARCH.cpp C/C++ Windows (CR+LF) WINDOWS-1252 Line 35, Col 1, Pos 766 Insert Read/Write default
29°C Rain showers 1:48 AM 9/24/2024
BST INSERT_DELETE_SEARCH.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global> search(struct node* root, int key, int& index, string path = "", int currentIndex = 0): node
Start here x BST INSERT_DELETE_SEARCH.cpp x
65
66 struct node* deleteNode(struct node* root, int t) {
67     if (root == NULL) {
68         cout << "Value " << t << " is not found for deletion.\n";
69         return NULL;
70     }
71
72     if (t < root->key) {
73         root->left = deleteNode(root->left, t);
74     } else if (t > root->key) {
75         root->right = deleteNode(root->right, t);
76     } else {
77
78         if (root->left == NULL) {
79             struct node* temp = root->right;
80             delete root;
81             return temp;
82         } else if (root->right == NULL) {
83             struct node* temp = root->left;
84             delete root;
85             return temp;
86         }
87
88     }
```

```
BST INSERT_DELETE_SEARCH.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global> main():int
Start here x BST INSERT_DELETE_SEARCH.cpp x
89     struct node* temp = findMin(root->right);
90     root->key = temp->key;
91     root->right = deleteNode(root->right, temp->key);
92 }
93 return root;
94 }
95
96 void displayMenu() {
97     cout << "Menu:\n";
98     cout << "1. Insert a node\n";
99     cout << "2. Search for a node\n";
100    cout << "3. Delete a node\n";
101    cout << "4. Display tree (pre-order)\n";
102    cout << "5. Exit\n";
103 }
104
105 int main() {
106     struct node* root = NULL;
107     int choice, value, index;
108
109     while (true) {
110         displayMenu();
111         cout << "Enter your choice: ";
112         cin >> choice;
```

E:\BST INSERT\_DELETE\_SEARCH.cpp C/C++ Windows (CR+LF) WINDOWS-1252 Line 109, Col 19, Pos 2647 Insert Read/Write default 1:49 AM 9/24/2024

```
BST INSERT_DELETE_SEARCH.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global> main():int
Start here x BST INSERT_DELETE_SEARCH.cpp x
104
105 int main() {
106     struct node* root = NULL;
107     int choice, value, index;
108
109     while (true) {
110         displayMenu();
111         cout << "Enter your choice: ";
112         cin >> choice;
113
114         switch (choice) {
115             case 1:
116                 cout << "Enter the value to insert: ";
117                 cin >> value;
118                 root = insertNode(root, value);
119                 cout << "Value inserted.\n";
120                 break;
121
122             case 2:
123                 cout << "Enter the value to search for: ";
124                 cin >> value;
125                 index = -1;
126                 if (search(root, value, index) != NULL) {
127                     // Output handled in search function
```

E:\BST INSERT\_DELETE\_SEARCH.cpp C/C++ Windows (CR+LF) WINDOWS-1252 Line 109, Col 19, Pos 2647 Insert Read/Write default 1:49 AM 9/24/2024

```
BST INSERT_DELETE_SEARCH.cpp - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
main(): int
122 case 2:
123     cout << "Enter the value to search for: ";
124     cin >> value;
125     index = -1;
126     if (search(root, value, index) != NULL) {
127         // Output handled in search function
128     } else {
129         cout << "Not Found\n";
130     }
131     break;
132
133 case 3:
134     cout << "Enter the value to delete: ";
135     cin >> value;
136     root = deleteNode(root, value);
137     break;
138
139 case 4:
140     cout << "Binary tree in pre-order:\n";
141     preOrder(root);
142     cout << endl;
143     break;
144
145 case 5:
```

## Output:

## Insert:

```
'E:\BST INSERT_DELETE_SEARCH' x + v
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 1
Enter the value to insert: 10
Value inserted.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 1
Enter the value to insert: 5
Value inserted.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 1
Enter the value to insert: 15
Value inserted.
Menu:
1. Insert a node
```

```
E:\BST_INSERT_DELETE_SEARCH >
Value inserted.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 1
Enter the value to insert: 3
Value inserted.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 1
Enter the value to insert: 7
Value inserted.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 1
Enter the value to insert: 12
Value inserted.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 1
Enter the value to insert: 18
Value inserted.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 4
Binary tree in pre-order:
10 5 3 7 15 12 18
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 2
Enter the value to search for: 15
Found at index: 1 (Path: right -> )
Menu:
1. Insert a node
```

## Search:

```
"E:\BST_INSERT_DELETE_SEARCH" x + v
10 5 3 7 15 12 18
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 2
Enter the value to search for: 15
Found at index: 1 (Path: right -> )
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 4
Binary tree in pre-order:
10 5 3 7 15 12 18
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 2
Enter the value to search for: 2
Not Found
Menu:
```

```
"E:\BST_INSERT_DELETE_SEARCH" x + v
5. Exit
Enter your choice: 2
Enter the value to search for: 2
Not Found
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 2
Enter the value to search for: 15
Found at index: 1 (Path: right -> )
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 2
Enter the value to search for: 5
Found at index: 1 (Path: left -> )
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 3
```

## Delete:

```
"E:\BST_INSERT_DELETE_SEARCH" x + v
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 3
Enter the value to delete: 10
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 4
Binary tree in pre-order:
12 5 3 7 15 18
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 3
Enter the value to delete: 2
Value 2 is not found for deletion.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 4
Binary tree in pre-order:
12 5 3 7 15 18
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 3
Enter the value to delete: 2
Value 2 is not found for deletion.
Menu:
1. Insert a node
2. Search for a node
3. Delete a node
4. Display tree (pre-order)
5. Exit
Enter your choice: 5
Exiting...

Process returned 0 (0x0)   execution time : 180.789 s
Press any key to continue.
```



### **Problem:03(Part-1): Discuss time complexity of insertion, deletion and search in a balanced BST.**

#### **Solution:**

Time complexity between BST insertion, deletion and search is given below:

Operation	Best Case	Average Case	Worst Case
Insertion	$O(1)$	$O(\log_2 n)$	$O(\log_2 n)$
Deletion	$O(1)$	$O(\log_2 n)$	$O(\log_2 n)$
Search	$O(1)$	$O(\log_2 n)$	$O(\log_2 n)$