

```

1  /*
2  给定二叉搜索树（BST）的根节点和一个值。 你需要在BST中找到节点值等于给定值的节点。 返回以
   该节点为根的子树。 如果节点不存在，则返回 NULL。
3
4  例如，
5
6  给定二叉搜索树：
7
8      4
9     / \
10    2   7
11   / \
12  1   3
13
14  和值：2
15
16  你应该返回如下子树：
17
18      2
19     / \
20    1   3
21
22  在上述示例中，如果要找的值是 5，但因为没有节点值为 5，我们应该返回 NULL。
23
24  来源：力扣（LeetCode）
25  链接：https://leetcode-cn.com/problems/search-in-a-binary-search-tree
26  著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
27  */

```

分析:

只要找到 $value = val$ 的节点即可

- 方法一:递归法
- 方法二:迭代法

方法一:C_递归法

```

1  /**
2   * Definition for a binary tree node.
3   * struct TreeNode {
4   *     int val;
5   *     struct TreeNode *left;
6   *     struct TreeNode *right;
7   * };
8   */
9
10
11
12  struct TreeNode* ret_val = NULL;
13
14
15  void __searchBSTNode(struct TreeNode* node, int val)
16  {

```

```

17     if(node==NULL)
18     {
19         return;
20     }
21
22     if (node->val == val)
23     {
24         ret_val = node;
25         return;
26     }
27     else if (node->val < val)
28     {
29         __searchBSTNode(node->right, val);
30     }
31     else
32     {
33         __searchBSTNode(node->left, val);
34     }
35
36
37 }
38
39 struct TreeNode* searchBST(struct TreeNode* root, int val)
40 {
41     ret_val = NULL;
42     __searchBSTNode(root, val);
43     return ret_val;
44 }
45
46 /*
47 执行结果:
48 通过
49 显示详情
50 执行用时 :64 ms, 在所有 C 提交中击败了9.70% 的用户
51 内存消耗 :16.4 MB, 在所有 C 提交中击败了100.00%的用户
52 */

```

方法二:C_迭代法

```

1  struct TreeNode* searchBST(struct TreeNode* root, int val)
2  {
3
4      struct TreeNode* temp = root;
5      while (1)
6      {
7          if(temp==NULL)
8          {
9              break;
10         }
11
12         if(temp->val< val)
13         {
14             temp = temp -> right;
15         }
16         else if(temp->val > val)
17         {

```

```

18         temp = temp -> left;
19     }
20     else
21     {
22         return temp;
23     }
24 }
25 return NULL;
26 }
27
28 /*
29 执行结果:
30 通过
31 显示详情
32 执行用时 :56 ms, 在所有 C 提交中击败了14.93% 的用户
33 内存消耗 :16.4 MB, 在所有 C 提交中击败了100.00%的用户
34 */
35
36
37
38 struct TreeNode* searchBST(struct TreeNode* root, int val)
39 {
40     while (1)
41     {
42         if(root==NULL)
43         {
44             break;
45         }
46         if(root->val< val)
47         {
48             root = root -> right;
49         }
50         else if(root->val > val)
51         {
52             root = root -> left;
53         }
54         else
55         {
56             return root;
57         }
58     }
59     return NULL;
60 }
61
62 /*
63 执行结果:
64 通过
65 显示详情
66 执行用时 :36 ms, 在所有 C 提交中击败了68.66% 的用户
67 内存消耗 :16.6 MB, 在所有 C 提交中击败了100.00%的用户
68 */

```