

LintCode领扣题解 (/problem) / 二叉搜索树中最接近的值 · Closest Binary Search Tree Value

二叉搜索树中最接近的值 · Closest Binary Search Tree Value

中文

[微软 \(/problem/?tags=microsoft\)](/problem/?tags=microsoft)[Snapchat \(/problem/?tags=snapchat\)](/problem/?tags=snapchat)[谷歌 \(/problem/?tags=google\)](/problem/?tags=google)[二叉查找树 \(/problem/?tags=binary-search-tree\)](/problem/?tags=binary-search-tree)

描述

给一棵非空二叉搜索树以及一个target值, 找到在BST中最接近给定值的节点值

① * 给出的目标值为浮点数 * 我们可以保证只有唯一一个最接近给定值的节点

样例

样例1

输入: root = {5,4,9,2,#,8,10} and target = 6.124780

输出: 5

解释:

二叉树 {5,4,9,2,#,8,10}, 表示如下的树结构:

```
    5
   / \
  4   9
 / \  / \
2  8 10
```

样例2

输入: root = {3,2,4,1} and target = 4.142857

输出: 4

解释:

二叉树 {3,2,4,1}, 表示如下的树结构:

```
    3
   / \
  2   4
 /
1
```

在线评测地址: <https://www.lintcode.com/problem/closest-binary-search-tree-value/> (<https://www.lintcode.com/problem/closest-binary-search-tree-value/>)

收起题目描述 ^

语言类型

ALL (50)

python (26)

java (17)

c++ (6)

golang (1)

邀请
有礼

invitation/shi

上传题解



Boolean

更新于 12/17/2020, 11:40:26 PM

非递归的版本 根据target的值与当前root.val的大小找到最接近target的两个值

```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
"""
Definition of TreeNode:
class TreeNode:
    def __init__(self, val):
        self.val = val
        self.left, self.right = None, None
"""

class Solution:
    """
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    """
    def closestValue(self, root, target):
        upper = root
        lower = root
        while root:
            if target > root.val:
                lower = root
                root = root.right
            elif target < root.val:
                upper = root
                root = root.left
            else:
                return root.val
        if abs(upper.val - target) <= abs(lower.val - target):
            return upper.val
        return lower.val

```

👍 获赞 31

💬 5 条评论

你的口袋题库

2000+ 算法真题、国内外名企题库免费开放



九章算法APP

令狐冲

更新于 11/28/2020, 12:32:10 AM

算法很简单, 求出 lowerBound 和 upperBound。即 $< \text{target}$ 的最大值和 $\geq \text{target}$ 的最小值。然后在两者之中去比较谁更接近, 然后返回即可。

时间复杂度为 $O(h)$, 注意如果你使用 in-order traversal 的化, 时间复杂度会是 $o(n)$ 并不是最优的。另外复杂度也不是 $O(\log n)$ 因为BST 并不保证树高是 $\log n$ 的。

```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code

```

```
*/
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     int val;
 *     TreeNode left;
 *     TreeNode right;
 *     TreeNode(int x) { val = x; }
 * }
 */
class Solution {
    public int closestValue(TreeNode root, double target) {
        if (root == null) {
            return 0;
        }

        TreeNode lowerNode = lowerBound(root, target);
        TreeNode upperNode = upperBound(root, target);

        if (lowerNode == null) {
            return upperNode.val;
        }

        if (upperNode == null) {
            return lowerNode.val;
        }

        if (target - lowerNode.val > upperNode.val - target) {
            return upperNode.val;
        }

        return lowerNode.val;
    }

    // find the node with the largest value that smaller than target
    private TreeNode lowerBound(TreeNode root, double target) {
        if (root == null) {
            return null;
        }

        if (target <= root.val) {
            return lowerBound(root.left, target);
        }

        // root.val < target
        TreeNode lowerNode = lowerBound(root.right, target);
        if (lowerNode != null) {
            return lowerNode;
        }

        return root;
    }

    // find the node with the smallest value that larger than or equal to target
    private TreeNode upperBound(TreeNode root, double target) {
        if (root == null) {
            return null;
        }

        if (root.val < target) {
            return upperBound(root.right, target);
        }

        // root.val >= target
        TreeNode upperNode = upperBound(root.left, target);
        if (upperNode != null) {
            return upperNode;
        }
    }
}
```

```
        return root;
    }
}
```

👍 获赞 20 💬 13 条评论



九章算法助教团队

更新于 12/12/2020, 7:10:15 PM

直接将整棵树遍历一遍，一边遍历一边更新最接近的值即可

```
/**
 * 本参考程序由九章算法用户提供。版权所有，转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作，授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括：九章算法班 2020升级版，算法强化班，算法基础班，北美算法面试高频题班，Java 高级工程师 P6+ 小班课，面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括：系统设计 System Design，面向对象设计 OOD
 * - 专题及项目类课程包括：动态规划专题班，Big Data - Spark 项目实战，Django 开发项目课
 * - 更多详情请见官方网站：http://www.jiuzhang.com/?utm_source=code
 */
class Solution {
public:
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    int closestValue(TreeNode * root, double target) {
        // write your code here
        int closest = root->val;
        while (root) {
            if (abs(closest - target) >= abs(root->val - target)) {
                closest = root->val;
            }
            root = target < root->val ? root->left : root->right;
        }
        return closest;
    }
};
```

👍 获赞 4 💬 1 条评论



令狐冲

更新于 9/8/2020, 8:24:55 PM

分别找到上边界和下边界，比一下就好了

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
"""
Definition of TreeNode:
class TreeNode:
    def __init__(self, val):
        self.val = val
        self.left, self.right = None, None
"""

class Solution:
    """
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    """
    def closestValue(self, root, target):
        if root is None:
            return None

        lower = self.get_lower_bound(root, target)
        upper = self.get_upper_bound(root, target)
        if lower is None:
            return upper.val
        if upper is None:
            return lower.val

        if target - lower.val < upper.val - target:
            return lower.val
        return upper.val

    def get_lower_bound(self, root, target):
        # get the largest node that < target
        if root is None:
            return None

        if target < root.val:
            return self.get_lower_bound(root.left, target)

        lower = self.get_lower_bound(root.right, target)
        return root if lower is None else lower

    def get_upper_bound(self, root, target):
        # get the smallest node that >= target
        if root is None:
            return None

        if target >= root.val:
            return self.get_upper_bound(root.right, target)

        upper = self.get_upper_bound(root.left, target)
        return root if upper is None else upper
```

👍 获赞 4

💬 5 条评论



r同学

更新于 12/16/2020, 10:59:20 AM

特别简单好理解的方法, 非递归: 如果当前root值比target大, 就暂且把这个root值当成上限, 然后往左边走 如果当前root值比target小, 就暂且把这个root值当成下限, 然后往右边走 左右摇摆着走, 知道发现两个最接近target的值, 由于是inplace的更新上下限, 而且不递归, 所以没有额外的空间损耗 $O(h)$ time and $O(1)$ space

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到到的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution:
    """
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    """
    def closestValue(self, root, target):
        # write your code here

        upper = root
        lower = root

        while root:
            if root.val > target:
                upper = root
                root = root.left
            elif root.val < target:
                lower = root
                root = root.right
            else:
                return root.val

        if abs(upper.val - target) > abs(lower.val - target):
            return lower.val
        return upper.val
```

👍 获赞 25

💬 6 条评论



ctc316

更新于 6/9/2020, 7:03:45 AM

使用分治法: 若 target 比 root 小, 往左子树找最大值與 root 比較和 target 的差距; target 比 root 大時, 則往右找最小值來比較。Time: $O(h)$, Space: $O(1)$

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
/**
 * Definition of TreeNode:
 * public class TreeNode {
 *     public int val;
 *     public TreeNode left, right;
 *     public TreeNode(int val) {
 *         this.val = val;
 *         this.left = this.right = null;
 *     }
 * }
 */

public class Solution {
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    public int closestValue(TreeNode root, double target) {
        if (root == null) {
            return Integer.MIN_VALUE;
        }

        if (target < root.val) {
            if (root.left != null) {
                int left = closestValue(root.left, target);
                if (Math.abs(left - target) < Math.abs(root.val - target)) {
                    return left;
                }
            }
        } else if (target > root.val) {
            if (root.right != null) {
                int right = closestValue(root.right, target);
                if (Math.abs(right - target) < Math.abs(root.val - target)) {
                    return right;
                }
            }
        }

        return root.val;
    }
}
```

👍 获赞 24

💬 4 条评论



r同学

更新于 6/9/2020, 7:03:45 AM

特别简单好理解的方法, 非递归: 如果当前root值比target大, 就暂且把这个root值当成上限, 然后往左边走 如果当前root值比target小, 就暂且把这个root值当成下限, 然后往右边走 左右摇摆着走, 知道发现两个最接近target的值, 由于是inplace的更新上下限, 而且不递归, 所以没有额外的空间损耗 $O(h)$ time and $O(1)$ space

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    public int closestValue(TreeNode root, double target) {
        // write your code here
        /**
         non-recursion method. find the upper and lower just by moving step by step. O(h) time and O(1) space
         recursion method: find upper and find lower two methods.
         */

        if (root == null){
            return 0;
        }

        TreeNode upper = root;
        TreeNode lower = root;

        while (root != null){
            if (root.val > target){
                upper = root;
                root = root.left;
            }else if (root.val < target){
                lower = root;
                root = root.right;
            }else {
                return root.val;
            }
        }

        if (Math.abs(upper.val - target) > Math.abs(target - lower.val)){
            return lower.val;
        }
        return upper.val;
    }
}
```

👍 获赞 22

💬 2 条评论

**YouXiuHaoChiMa**

更新于 6/9/2020, 7:03:46 AM

用非递归实现的upperbound 和 lowerbound. 方法是向接近target的分支遍历,直到None 如果是找upperbound, 那么向左走时记录last node(这是这是当前最接近target的大数); 如果是找lowerbound, 那么向右走时记录last node(因为这是当前最接近target的小数)


```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
"""

Definition of TreeNode:
class TreeNode:
    def __init__(self, val):
        self.val = val
        self.left, self.right = None, None
"""

class Solution:
    """
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    """
    def closestValue(self, root, target):
        if not root:
            raise Exception

        lower = self.findLower(root, target)
        upper = self.findUpper(root, target)
        if lower and upper:
            if target - lower.val > upper.val - target:
                return upper.val
            else:
                return lower.val
        if lower:
            return lower.val
        if upper:
            return upper.val

    def findLower(self, root, target):
        curr = root
        last = None
        while curr:
            if curr.val <= target:
                last = curr
                curr = curr.right
            else:
                curr = curr.left
        return last # may be None if no one <= target

    def findUpper(self, root, target):
        curr = root
        last = None
        while curr:
            if curr.val > target:
                last = curr
                curr = curr.left
            else:
                curr = curr.right
        return last # may be None if no one > target
```

👍 获赞 6

💬 添加评论

**九章用户T12FVG**

更新于 12/18/2020, 11:03:13 PM

while 循环一直往下走, 直到底部, 每次比较和更新当前 root

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
/**
 * Definition of TreeNode:
 * public class TreeNode {
 *     public int val;
 *     public TreeNode left, right;
 *     public TreeNode(int val) {
 *         this.val = val;
 *         this.left = this.right = null;
 *     }
 * }
 */

public class Solution {
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    public int closestValue(TreeNode root, double target) {
        // write your code here
        int ans = root.val;
        double diff = Double.MAX_VALUE;

        while (root != null) {
            double newDiff = Math.abs(target - root.val);
            if (newDiff < diff) {
                ans = root.val;
                diff = newDiff;
            }
            if (target > root.val) {
                root = root.right;
            } else {
                root = root.left;
            }
        }
        return ans;
    }
}
```

👍 获赞 5

💬 添加评论

**Tina**

更新于 10/20/2020, 1:55:48 AM

non-recursion

- 把精选里面的递归改成了while循环, 其他思路跟参考答案一致

```
/**
 * 本参考程序由九章算法用户提供。版权所有，转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作，授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括：九章算法班 2020升级版，算法强化班，算法基础班，北美算法面试高频题班，Java 高级工程师 P6+ 小班课，面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括：系统设计 System Design，面向对象设计 OOD
 * - 专题及项目类课程包括：动态规划专题班，Big Data - Spark 项目实战，Django 开发项目课
 * - 更多详情请见官方网站：http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    public int closestValue(TreeNode root, double target) {
        if (root == null) {
            return -1;
        }
        TreeNode lower = lowerBound(root, target);
        TreeNode upper = upperBound(root, target);

        if (lower == null) {
            return upper.val;
        }
        if (upper == null) {
            return lower.val;
        }

        if (target - lower.val < upper.val - target) {
            return lower.val;
        }
        return upper.val;
    }

    private TreeNode lowerBound(TreeNode root, double target) {
        TreeNode lower = null;
        while (root != null) {
            if (target >= root.val) {
                lower = root;
                root = root.right;
            } else {
                root = root.left;
            }
        }
        return lower;
    }

    private TreeNode upperBound(TreeNode root, double target) {
        TreeNode upper = null;
        while (root != null) {
            if (target > root.val) {
                root = root.right;
            } else {
                upper = root;
                root = root.left;
            }
        }
        return upper;
    }
}
```

👍 获赞 3

💬 添加评论

**Julio@LintCode**

更新于 6/9/2020, 7:03:47 AM

简单的分治法 不需要那么多逻辑, 利用BST的比较即可 "" @param root: the given BST @param target: the given target @return: the value in the BST that is closest to the target ""

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution:
    ""
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    ""
    def closestValue(self, root, target):
        # write your code here
        if not root:
            return sys.maxsize

        if root.val == target:
            return target

        elif target > root.val:
            rClose = self.closestValue(root.right, target)
            if abs(root.val - target) < abs( rClose - target):
                return root.val
            else:
                return rClose
        else:
            lClose = self.closestValue(root.left, target)
            if abs(root.val - target) < abs( lClose - target):
                return root.val
            else:
                return lClose
```

👍 获赞 3

💬 添加评论

**CodingMrWang**

更新于 8/29/2020, 3:58:07 PM

要么是当前root, 要么去left或者right找。函数的定义是找到最小的。所以拿向左或者向右找的结果和root比, 返回小的那个

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
public int closestValue(TreeNode root, double target) {
    // write your code here
    double curdis = Math.abs(root.val - target);
    int val = 0;
    if (target > root.val) {
        if (root.right == null) {
            return root.val;
        }
        val = closestValue(root.right, target);
    } else {
        if (root.left == null) {
            return root.val;
        }
        val = closestValue(root.left, target);
    }
    if (Math.abs(val - target) > curdis) {
        return root.val;
    } else {
        return val;
    }
}
```

👍 获赞 2

💬 添加评论

**九章用户LUTP67**

更新于 8/28/2020, 10:34:26 PM

Time Complexity is O(h) same as the answer provided by jiuzhang. Shorter Code.

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
/**
 * Definition of TreeNode:
 * public class TreeNode {
 *     public int val;
 *     public TreeNode left, right;
 *     public TreeNode(int val) {
 *         this.val = val;
 *         this.left = this.right = null;
 *     }
 * }
 */

public class Solution {
    TreeNode closestNode;

    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    public int closestValue(TreeNode root, double target) {
        if (root == null) {
            return closestNode.val;
        }

        if (closestNode == null) {
            closestNode = root;
        } else if (Math.abs(closestNode.val - target) > Math.abs(root.val - target)) {
            closestNode = root;
        }

        if (target < root.val) {
            return closestValue(root.left, target);
        } else {
            return closestValue(root.right, target);
        }
    }
}
```

👍 获赞 2 💬 添加评论



NingLee

更新于 6/9/2020, 7:03:50 AM

时间复杂度, 也是 $O(H)$ H 是BST树的高度。思想就是按照BST的性质去查找, 并不断的更新记录比target小的node、比target大的node。遍历的过程如果遇到target就返回, 否则最后比较下记录的两个节点, 哪个近就返回哪个

```
/**
 * 本参考程序由九章算法用户提供。版权所有，转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作，授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括：九章算法班 2020升级版，算法强化班，算法基础班，北美算法面试高频题班，Java 高级工程师 P6+ 小班课，面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括：系统设计 System Design，面向对象设计 OOD
 * - 专题及项目类课程包括：动态规划专题班，Big Data - Spark 项目实战，Django 开发项目课
 * - 更多详情请见官方网站：http://www.jiuzhang.com/?utm_source=code
 */
int closestValue(TreeNode * root, double target) {
    if (root == NULL) {
        return -1;
    }

    TreeNode *lessNode = root;
    TreeNode *biggerNode = root;
    TreeNode *curt = root;

    while (curt != NULL) {
        if (curt->val == target) {
            return target;
        } else if (curt->val > target) {
            biggerNode = curt;
            curt = curt->left;
        } else {
            lessNode = curt;
            curt = curt->right;
        }
    }

    if (abs(lessNode->val - target) < abs(biggerNode->val - target)) {
        return lessNode->val;
    }

    return biggerNode->val;
}
```

👍 获赞 2

💬 添加评论



九章用户ZWI2PQ

更新于 6/9/2020, 7:03:50 AM

如果target小于root，那么右子树的结点不可能比root更接近target，所以只需记住当前root为最接近的node，并遍历左子树并比较左子结点是否比当前最近结点更接近target，如果是则update最近结点。反之则遍历右子树。

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
public class Solution {
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    private int result;
    public int closestValue(TreeNode root, double target) {
        // write your code here
        if (root == null) return -1;
        result = root.val;
        traverse(root, target);
        return result;
    }

    private void traverse(TreeNode root, double target) {
        if (root == null) return;
        if (target - root.val == 0.0) {
            result = root.val;
            return;
        }

        if (Math.abs(target - root.val) < Math.abs(target - result)) {
            result = root.val;
        }
        if (target < root.val) {
            traverse(root.left, target);
        } else {
            traverse(root.right, target);
        }
    }
}
```

👍 获赞 2 💬 添加评论



华山扫地僧

更新于 6/9/2020, 7:03:50 AM

使用的non-recursion的方法, 有点像层级遍历, 然后打擂台, 感觉比较其他解答更加直观一点。


```
/**
 * 本参考程序由九章算法用户提供。版权所有，转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作，授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括：九章算法班 2020升级版，算法强化班，算法基础班，北美算法面试高频题班，Java 高级工程师 P6+ 小班课，面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括：系统设计 System Design，面向对象设计 OOD
 * - 专题及项目类课程包括：动态规划专题班，Big Data - Spark 项目实战，Django 开发项目课
 * - 更多详情请见官方网站：http://www.jiuzhang.com/?utm_source=code
 */
class Solution:
    """
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    """
    def closestValue(self, root, target):

        if root is None :
            return

        min_val, min_node = sys.maxsize, root
        stack = [root]

        while stack :
            node = stack.pop()
            if abs(node.val - target) <= min_val :
                min_val = abs(node.val - target)
                mini_node = node
            if node.left is not None :
                stack.append(node.left)
            if node.right is not None :
                stack.append(node.right)

        return mini_node.val
```

👍 获赞 2 💬 添加评论



Su

更新于 9/2/2020, 11:04:01 PM

BST中，左子树中所有结点一定不会比根结点小，右子树中所有结点一定不会比根结点小。利用这个特性，我们遍历结点时就可以砍掉一半的路径：

- 若target比当前子树的根结点小：那么目标值肯定不在当前子树的左子树中，因为比起当前子树的左子树中任意一点，当前子树的根结点都要更接近target（就算当前子树的左子树中存在某个点和当前子树的根结点一样，那也相当于是与根结点比较了，没意义），肯定要么是从之前记录的某个点、当前子树的根结点、当前子树的右子树中的某个点当中选。
- 若target比当前子树的根结点小：同理。

因此就可以记录一个全局状态，利用BST的特性和上述描述进行遍历，每次仅搜索左右子树中的一棵，遍历过程中更新该状态。遍历完成后即可找到所求。

```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
"""
Definition of TreeNode:
class TreeNode:
    def __init__(self, val):
        self.val = val
        self.left, self.right = None, None
"""

class Solution:
    """
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    """
    def closestValue(self, root, target):
        self.closet = root
        self._search_bst(root, target)
        return self.closet.val

    def _search_bst(self, root, target):
        if not root:
            return

        if (target < root.val):
            self._search_bst(root.left, target)

        if abs(root.val - target) < abs(self.closet.val - target):
            self.closet = root

        if (target > root.val):
            self._search_bst(root.right, target)

```

👍 获赞 1 💬 添加评论



Angela

更新于 8/29/2020, 3:08:54 PM

<=> find TreeNode = target in BST 模版 + 改良recursion带bound = root, 即, 中头彩, 恭喜你; 知结果在左子树, 改上限; 知结果在右子树, 改下限;

```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
结束时刻, corner cases:
1) target < 最小值, 送走上限;
2) target > 最大值, 送走下限;
3) clarify: diff(target - 下限) = diff(上限 - target) 时, 谁算更近水楼台先得月?

```

思路来源, [狠狠抄袭] 令狐老师的精选代码, 感谢附上时间复杂度分析 $O(h)$, $h = \text{BST.height}$.

```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
/**
 * Definition of TreeNode:
 * public class TreeNode {
 *     public int val;
 *     public TreeNode left, right;
 *     public TreeNode(int val) {
 *         this.val = val;
 *         this.left = this.right = null;
 *     }
 * }
 */

public class Solution {
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    public int closestValue(TreeNode root, double target) {
        // Exception
        if (root == null) {
            return 0;
        }

        // find closest Node in BST.root, given target value
        // max{lowerNode} <= target <= min{upperNode}
        TreeNode result = helper(root, target, null, null);
        return result.val;
    }

    // find closest Node in BST.root, given target value
    // lowerBound = max{lowerNode}, upperBound = min{upperNode}
    private TreeNode helper(TreeNode root, double target,
                            TreeNode lowerBound, TreeNode upperBound) {
        // Termination
        if (root == null) {
            if (lowerBound == null) { // target < min in BST
                return upperBound;
            }
            if (upperBound == null) { // target > max IN BST
                return lowerBound;
            }
            return target - lowerBound.val <= upperBound.val - target ?
                lowerBound :
                upperBound;
        }

        TreeNode closestNode = null;
        if (root.val == target) {
            return root;
        } else if (root.val > target) {
            // search leftSubTree & update upperBound
            closestNode = helper(root.left, target, lowerBound, root);
        } else { //root.val < target
            // search rightSubTree & update lowerBound
            closestNode = helper(root.right, target, root, upperBound);
        }
        return closestNode;
    }
}

```

```
}
}
```

👍 获赞 1 💬 添加评论



Tommy

更新于 6/9/2020, 7:03:57 AM

令狐老师解法的C++版 令狐冲: “算法很简单, 求出 lowerBound 和 upperBound。即 $< \text{target}$ 的最大值和 $\geq \text{target}$ 的最小值。然后在两者之中去比较谁更接近, 然后返回即可。

时间复杂度为 $O(h)$, 注意如果你使用 in-order traversal 的话, 时间复杂度会是 $o(n)$ 并不是最优的。另外复杂度也不是 $O(\log n)$ 因为BST 并不保证树高是 $\log n$ 的。”

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution {
public:
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    int closestValue(TreeNode* root, double target) {
        if (NULL == root) {
            return 0;
        }

        TreeNode* lowerNode = lowerBound(root, target);
        TreeNode* upperNode = upperBound(root, target);

        if (NULL == lowerNode) {
            return upperNode->val;
        }

        if (NULL == upperNode) {
            return lowerNode->val;
        }

        if (target - lowerNode->val > upperNode->val - target) {
            return upperNode->val;
        }

        return lowerNode->val;
    }

    //return node->val < target
    TreeNode* lowerBound(TreeNode* root, double target) {
        if (NULL == root) {
            return NULL;
        }

        if (target <= root->val) {
            return lowerBound(root->left, target);
        }

        TreeNode* lowerNode = lowerBound(root->right, target);
```

```
        if (NULL == lowerNode) {
            return root;
        }
        return lowerNode;
    }

    TreeNode* upperBound(TreeNode* root, double target) {
        if (NULL == root) {
            return NULL;
        }

        if (target > root->val) {
            return upperBound(root->right, target);
        }

        TreeNode* upperNode = upperBound(root->left, target);
        if (NULL == upperNode) {
            return root;
        }
        return upperNode;
    }
};
```

👍 获赞 1

💬 2 条评论



九章用户PJGZHC

更新于 6/9/2020, 7:03:57 AM

Coming from Java version provided by Jiuzhang.

```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
"""
Definition of TreeNode:
class TreeNode:
    def __init__(self, val):
        self.val = val
        self.left, self.right = None, None
"""

class Solution:
    """
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    """
    def closestValue(self, root, target):
        # write your code here
        if root is None:
            return
        lowerBound = self.lower(root, target)
        upperBound = self.upper(root, target)

        if lowerBound is None:
            return upperBound.val

        if upperBound is None:
            return lowerBound.val

        if abs(lowerBound.val - target) < abs(upperBound.val - target):
            return lowerBound.val
        else:
            return upperBound.val

    def lower(self, root, target):
        if root is None:
            return None

        if root.val > target:
            return self.lower(root.left, target)

        lowerNode = self.lower(root.right, target)
        if lowerNode:
            return lowerNode

        return root

    def upper(self, root, target):
        if root is None:
            return None

        if root.val < target:
            return self.upper(root.right, target)

        upperNode = self.upper(root.left, target)
        if upperNode:
            return upperNode

        return root

```

👍 获赞 1 💬 添加评论



九章用户3C61RO

更新于 6/9/2020, 7:03:56 AM

思路是使用分治法: 若 target 比 root 小, 往左子树找最大值与 root 比较和 target 的差距; target 比 root 大时, 则往右找最小值来比较。非递归的实现是用minDiff变量记录便利路径上的最小差距。Time: O(h), Space: O(1)

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution {
public:
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    int closestValue(TreeNode* root, double target) {
        TreeNode* min = nullptr;
        double minDiff = std::numeric_limits<double>::max();

        while (root != nullptr) {
            double diff = abs(root->val - target);
            if (diff < minDiff) {
                minDiff = diff;
                min = root;
            }

            if (root->val == target) {
                break;
            }

            if (target < root->val) {
                root = root->left;
            } else {
                root = root->right;
            }
        }

        return (min == nullptr) ? 0 : min->val;
    }
};
```

👍 获赞 1 💬 添加评论



wvua

更新于 6/9/2020, 7:03:55 AM

Time complexity O(h), h denotes the height of the tree.

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
/**
 * Definition of TreeNode:
 * public class TreeNode {
 *     public int val;
 *     public TreeNode left, right;
 *     public TreeNode(int val) {
 *         this.val = val;
 *         this.left = this.right = null;
 *     }
 * }
 */

public class Solution {
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    public int closestValue(TreeNode root, double target) {
        // should not be null
        if (root == null) {
            return -1;
        }

        return dfs(root, target).val;
    }

    private TreeNode dfs(TreeNode node, double target) {
        if (node == null) {
            return null;
        }

        if (node.val == target) {
            return node;
        }

        TreeNode closest;
        if (node.val < target) {
            closest = dfs(node.right, target);
        } else {
            closest = dfs(node.left, target);
        }

        if (closest == null) {
            return node;
        }

        return Math.abs(target - closest.val) < Math.abs(target - node.val) ? closest : node;
    }
}
```

👍 获赞 1 💬 添加评论



The Art of Racing in the Rain

更新于 6/9/2020, 7:03:52 AM

每次只需要取一个node。O(H)时间复杂度, O(1)空间复杂度。LeetCode和LintCode上速度都是faster than 100%。

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
"""
Definition of TreeNode:
class TreeNode:
    def __init__(self, val):
        self.val = val
        self.left, self.right = None, None
"""

class Solution:
    """
    @param root: the given BST
    @param target: the given target
    @return: the value in the BST that is closest to the target
    """
    def closestValue(self, root, target):
        # write your code here
        if not root or not target:
            return -1

        res = sys.maxsize

        while root:
            if abs(root.val - target) < abs(res - target):
                res = root.val

            if root.val < target:
                root = root.right
            elif root.val > target:
                root = root.left
            else:
                return root.val
        return res
```

👍 获赞 1 💬 添加评论



Jet

更新于 6/9/2020, 7:03:52 AM

1)法1 中序遍历+二分 O(n) 贼慢 2)法2 upper bound, lower bound 若是平衡BST O(h) = O(logn) 贼快

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
/**
 * Definition of TreeNode:
 * class TreeNode {
```

```

* public:
*     int val;
*     TreeNode *left, *right;
*     TreeNode(int val) {
*         this->val = val;
*         this->left = this->right = NULL;
*     }
* }
*/
class Rtype{
public:
    int closest;
    TreeNode* node;
    Rtype(TreeNode* _node):node(_node){

    }

};

class Solution {
public:
    TreeNode* upper;
    TreeNode* lower;
    /**
     * @param root: the given BST
     * @param target: the given target
     * @return: the value in the BST that is closest to the target
     */
    int closestValue(TreeNode * root, double target) {
        upper = root;
        lower = root;
        help(root, target);
        if (abs(upper->val - target) <= abs(lower->val - target)) {
            return upper->val;
        }
        return lower->val;
    }

    void help(TreeNode* root, double target) {
        if (root == nullptr) {
            return;
        }
        if (root->val < target) {
            lower = root;
            help(root->right, target);
            return;
        }
        if (root->val > target) {
            upper = root;
            help(root->left, target);
            return;
        }
        return;
    }

    //O(n)
    // int closestValue(TreeNode * root, double target) {
    //     vector<TreeNode*> nodes;
    //     traverse(root, nodes);
    //     int res_index = binarySearch(nodes, 0, nodes.size() - 1, target);
    //     return nodes[res_index]->val;
    //     return 0;
    // }

    // int binarySearch(vector<TreeNode*>& nodes, int start, int end, double target) {
    //     int middle;
    //     while (start + 1 < end) {

```

```

//      middle = start + (end - start) / 2;
//      if (nodes[middle]->val <= target) {
//          start = middle;
//          continue;
//      }
//      end = middle;
//  }
//  if (abs(nodes[start]->val - target) <= abs(nodes[end]->val - target)) {
//      return start;
//  }
//  return end;
// }

// void traverse(TreeNode* root, vector<TreeNode*>& nodes) {
//     if (root == nullptr) {
//         return;
//     }
//     traverse(root->left, nodes);
//     nodes.push_back(root);
//     traverse(root->right, nodes);
// }
//first do
// int closestValue(TreeNode * root, double target) {
//     // write your code here


//     Rtype result=divideCon(root,target);
//     return result.closest;
// }

// Rtype divideCon(TreeNode* node, double target){
//     Rtype r(nullptr);
//     if(node==nullptr){
//         return r;
//     }
//     Rtype left = divideCon(node->left,target);
//     Rtype right = divideCon(node->right,target);
//     if(left.node==nullptr&&right.node==nullptr){
//         r.node=node;
//         r.closest=node->val;
//         return r;
//     }
//     if(left.node==nullptr){
//         //right !=
//         if( abs(target- right.closest )<abs(target-node->val) ){
//             r.node= right.node;
//             r.closest=right.closest;
//             return r;
//         }
//         r.node=node;
//         r.closest=node->val;
//         return r;
//     }
//     if(right.node==nullptr){
//         if( abs(target- left.closest )<abs(target-node->val) ){
//             r.node= left.node;
//             r.closest=left.closest;
//             return r;
//         }
//         r.node=node;
//         r.closest=node->val;
//         return r;
//     }
//     }

//     r.node=node;
//     r.closest=node->val;
//     if(abs(left.closest-target)<abs(r.closest-target)){
//         r.closest=left.closest;
//     }
// }

```

```
//      if(abs(right.closest-target)<abs(r.closest-target)){  
//          r.closest=right.closest;  
//      }  
//      return r;  
  
// }  
  
};
```

 获赞 1 添加评论[加载更多题解](#)

进阶课程

[视频+互动](#)[直播+互动](#)[直播+互动](#)[互动课](#)

九章算法班 2021 版

8周时间精通 57 个核心高频考点, 9 招击破 FLAG、BATJ 算法面试。22....

系统架构设计 System Design 2021 版

成为百万架构师必上。30 课时带你快速掌握 18 大系统架构设计知识点与面...

九章算法面试高频题冲刺班

每期更新 15% 题目, 考前押题, 一举拿下 FLAG & BATJ Offer

面向对象设计 OOD

应届生及亚马逊面试必考, IT 求职必备基础

[首页 \(/?skip_redirect=true\)](#) | [联系我们 \(mailto:info@jiuzhang.com\)](mailto:info@jiuzhang.com) | [加入我们 \(/joinus\)](#)

Copyright © 2013-2020 九章算法 浙ICP备19045946号-1
(<http://www.miibeian.gov.cn/>)

商务合作: [fukesu@jiuzhang.com \(mailto:fukesu@jiuzhang.com\)](mailto:fukesu@jiuzhang.com)



(<http://weibo.com/ninechapter>)



(<https://www.zhihu.com/people/crackinterview/>)

(/)