Java

```java
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     int val;
 *     TreeNode left;
 *     TreeNode right;
 *     TreeNode() {}
 *     TreeNode(int val) { this.val = val; }
 *     TreeNode(int val, TreeNode left, TreeNode right) {
 *         this.val = val;
 *         this.left = left;
 *         this.right = right;
 *     }
 * }
 */
class FindElements {

    public FindElements(TreeNode root) {

    }

    public boolean find(int target) {

    }
}

/**
 * Your FindElements object will be instantiated and called as such:
 * FindElements obj = new FindElements(root);
 * boolean param_1 = obj.find(target);
 */
```

JavaScript

```
/**
 * Definition for a binary tree node.
 * function TreeNode(val, left, right) {
 *     this.val = (val===undefined ? 0 : val)
 *     this.left = (left===undefined ? null : left)
 *     this.right = (right===undefined ? null : right)
 * }
 */
/**
 * @param {TreeNode} root
 */
var FindElements = function(root) {

};

/**
 * @param {number} target
 * @return {boolean}
 */
FindElements.prototype.find = function(target) {

};

/**
 * Your FindElements object will be instantiated and called as such:
 * var obj = new FindElements(root)
 * var param_1 = obj.find(target)
 */
```

TypeScript

```
/**
 * Definition for a binary tree node.
 * class TreeNode {
 *     val: number
 *     left: TreeNode | null
 *     right: TreeNode | null
 *     constructor(val?: number, left?: TreeNode | null, right?: TreeNode | null) {
 *         this.val = (val===undefined ? 0 : val)
 *         this.left = (left===undefined ? null : left)
 *         this.right = (right===undefined ? null : right)
 *     }
 * }
 */

class FindElements {
    constructor(root: TreeNode | null) {

    }

    find(target: number): boolean {

    }
}

/**
 * Your FindElements object will be instantiated and called as such:
 * var obj = new FindElements(root)
 * var param_1 = obj.find(target)
 */
```

C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode() : val(0), left(nullptr), right(nullptr) {}
 *     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
 *     TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left), right(right) {}
 * };
 */
class FindElements {
public:
    FindElements(TreeNode* root) {
        
    }
    
    bool find(int target) {
        
    }
};

/**
 * Your FindElements object will be instantiated and called as such:
 * FindElements* obj = new FindElements(root);
 * bool param_1 = obj->find(target);
 */
--------------------------------------------------------------------
```

C#

```
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     public int val;
 *     public TreeNode left;
 *     public TreeNode right;
 *     public TreeNode(int val=0, TreeNode left=null, TreeNode right=null) {
 *         this.val = val;
 *         this.left = left;
 *         this.right = right;
 *     }
 * }
 */
public class FindElements {

    public FindElements(TreeNode root) {

    }

    public bool Find(int target) {

    }
}

/**
 * Your FindElements object will be instantiated and called as such:
 * FindElements obj = new FindElements(root);
 * bool param_1 = obj.Find(target);
 */
-----------------------------------------------------------------------
```

Kotlin

```kotlin
/**
 * Example:
 * var ti = TreeNode(5)
 * var v = ti.`val`
 * Definition for a binary tree node.
 * class TreeNode(var `val`: Int) {
 *     var left: TreeNode? = null
 *     var right: TreeNode? = null
 * }
 */
class FindElements(root: TreeNode?) {

    fun find(target: Int): Boolean {

    }

}

/**
 * Your FindElements object will be instantiated and called as such:
 * var obj = FindElements(root)
 * var param_1 = obj.find(target)
 */
--------------------------------------------------------------------
```

Go

```go
/**
 * Definition for a binary tree node.
 * type TreeNode struct {
 *     Val int
 *     Left *TreeNode
 *     Right *TreeNode
 * }
 */
type FindElements struct {

}


func Constructor(root *TreeNode) FindElements {

}


func (this *FindElements) Find(target int) bool {

}


/**
 * Your FindElements object will be instantiated and called as such:
 * obj := Constructor(root);
 * param_1 := obj.Find(target);
 */
--------------------------------------------------------------------
```