Data Structures and Object Oriented Programming

Lecture 13

Dr. Naveed Anwar Bhatti

Webpage: naveedanwarbhatti.github.io

Object-Oriented Programming in C++

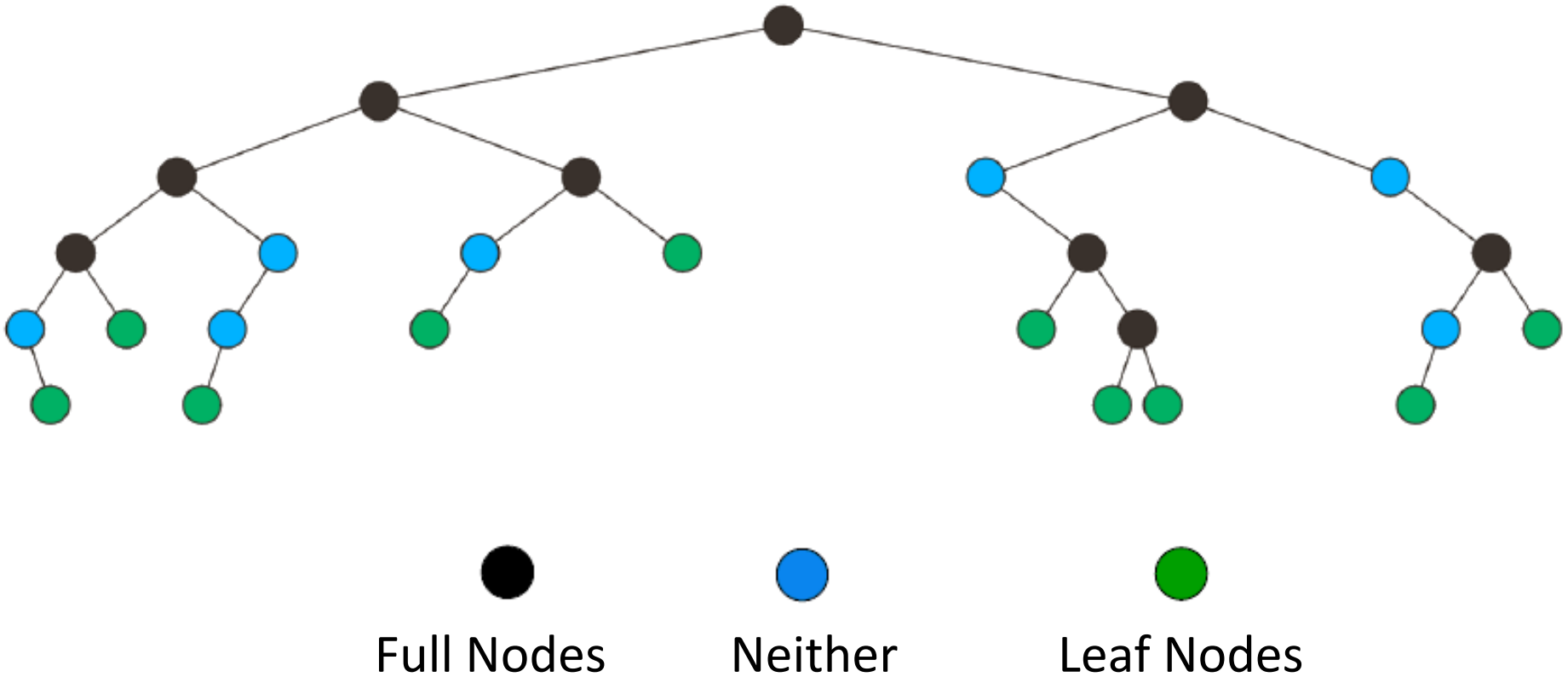
Binary Search Trees

`Insert()` and `Find()`

BT (Binary Tree) – Some Definitions

Full Node

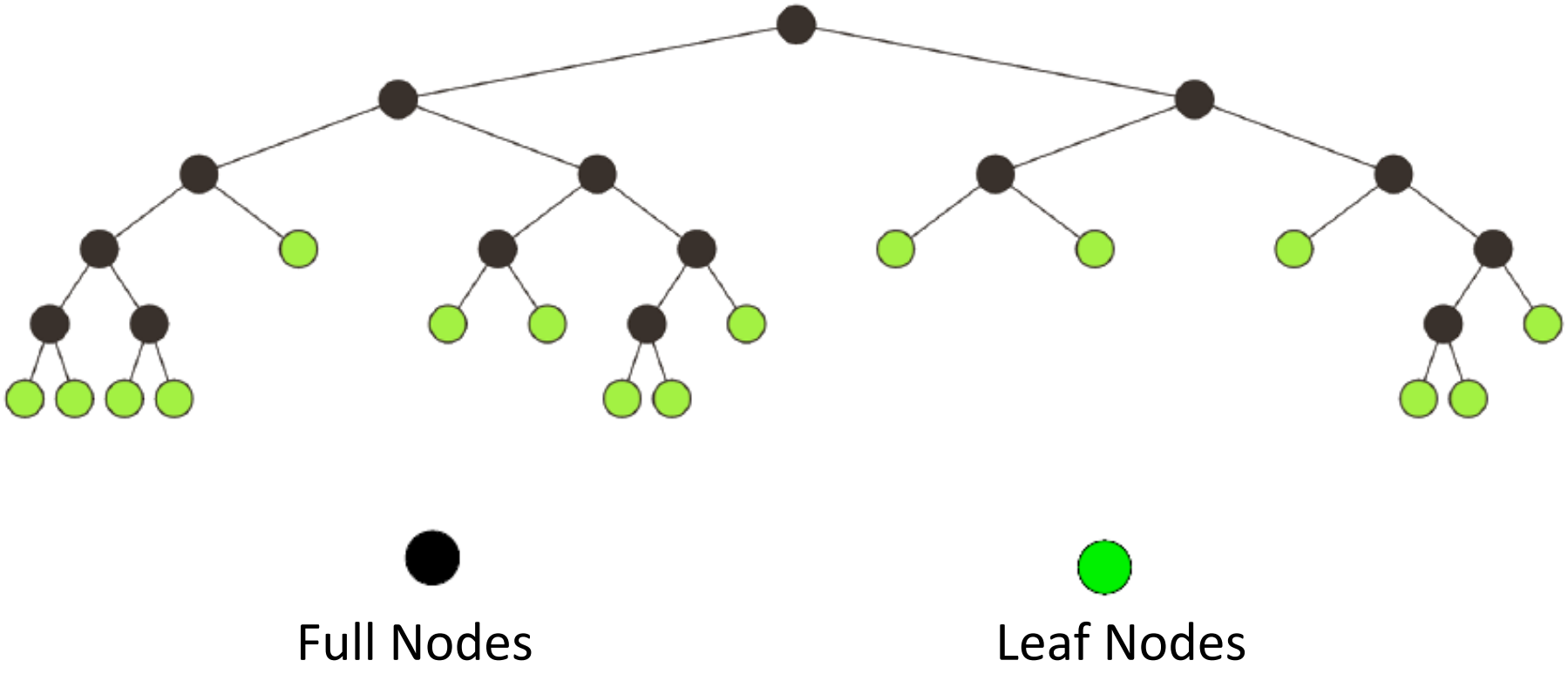
Definition: A full node is a node where both the left and right sub trees are non-empty trees



BT (Binary Tree) – Some Definitions

Full Binary Tree

Definition: A full binary tree is where each node either full node or leaf node



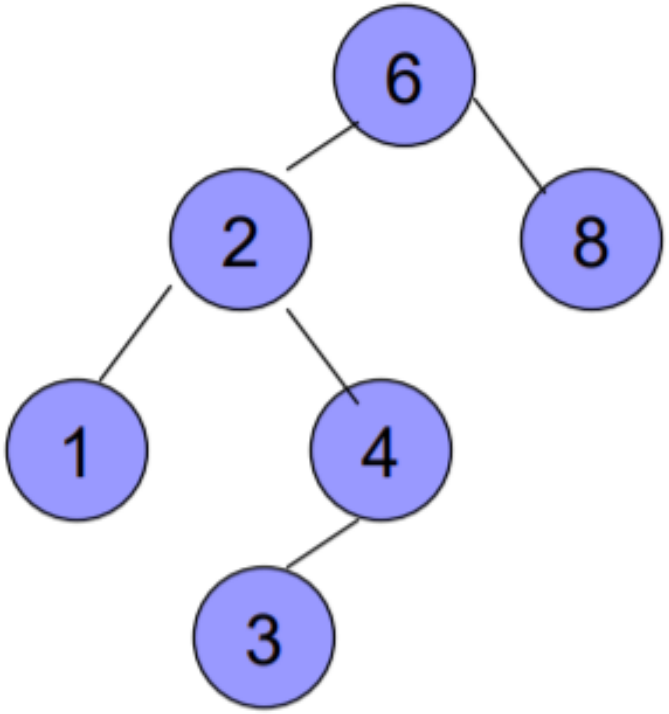
BST (Binary Search Tree) - Definition

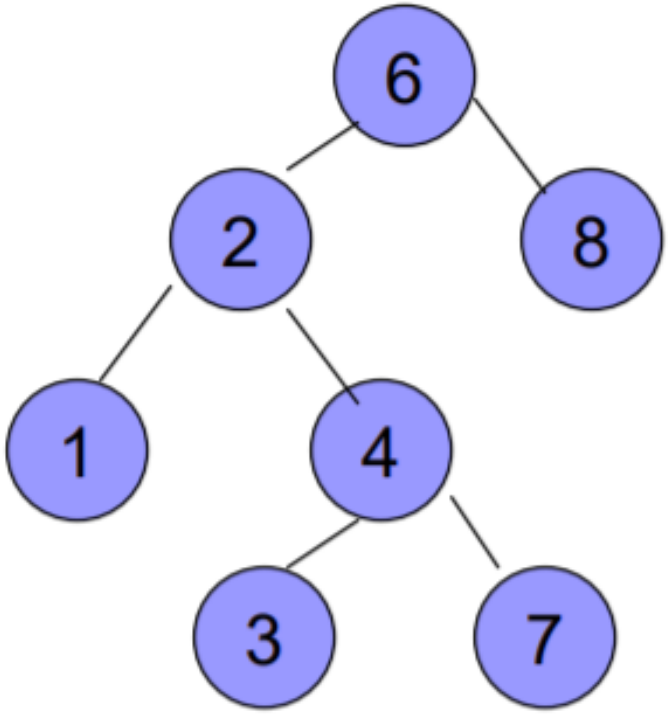
Also known as ***Totally Ordered Tree***

Definition: A binary tree **B** is called a binary search tree iff:

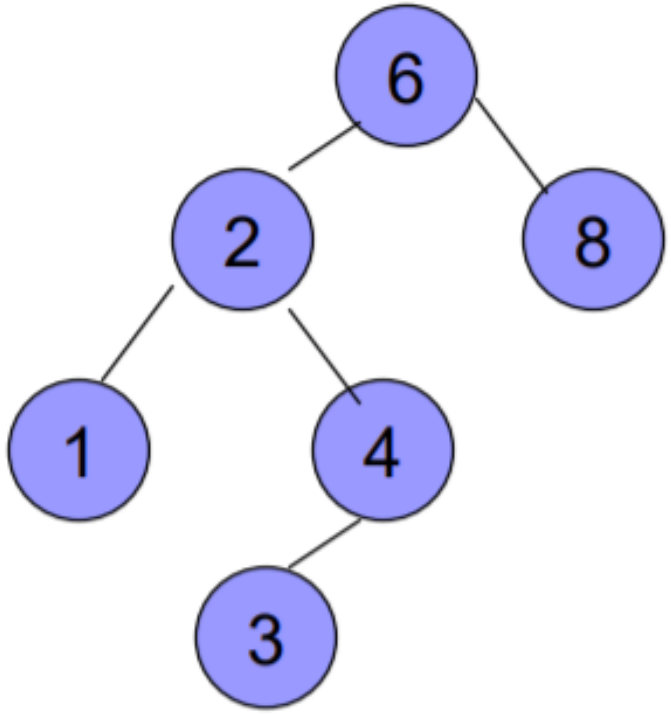
- For every node **T** in the **B** tree, the values of all the items in its ***left*** subtree are ***smaller*** than the item in **T**
- The values of all the items in its ***right*** subtree are ***larger*** than the item in **T**

Practice: are these BSTs?

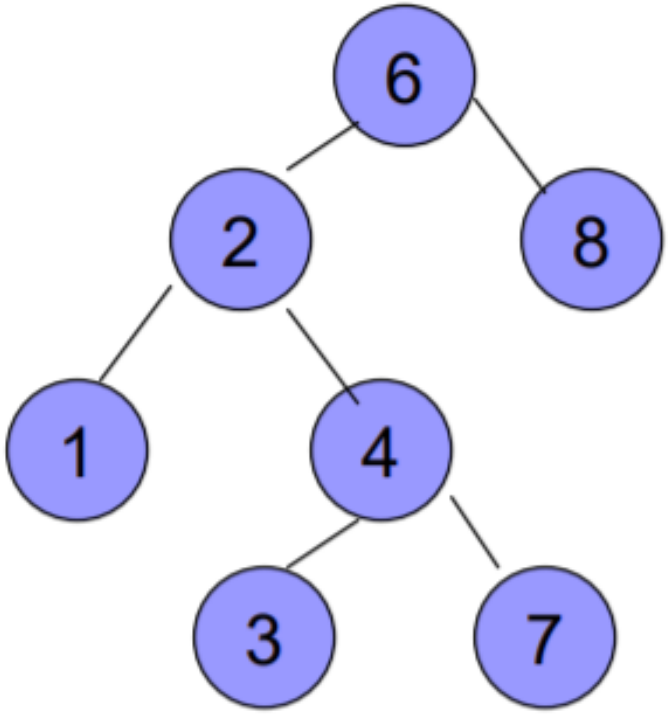




Practice: are these BSTs?



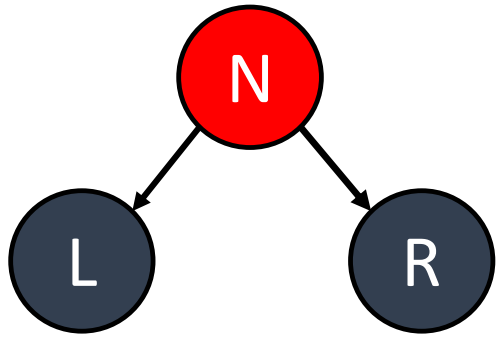
Binary Search Tree



Binary Tree

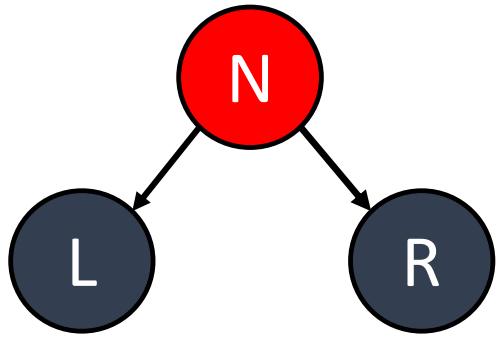
BST (Binary Search Tree) - Implementation

```
struct Node
{
    int data;
    Node* left = NULL;
    Node* right = NULL;
};
```



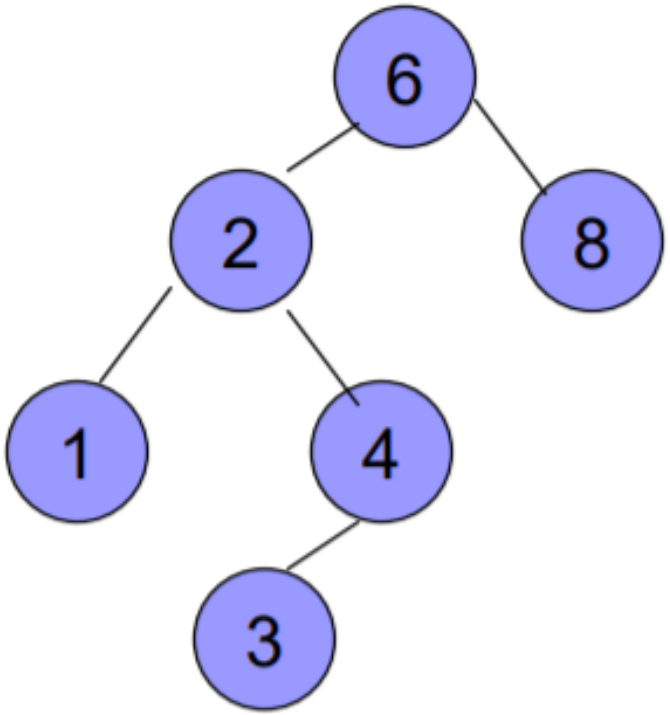
BST (Binary Search Tree) - Implementation

```
struct Node
{
    int data;
    Node* left = NULL;
    Node* right = NULL;
};

class BSTree
{

    Node* root = NULL;

    public:
        void insert(int value);
        bool find(int a);

};
```



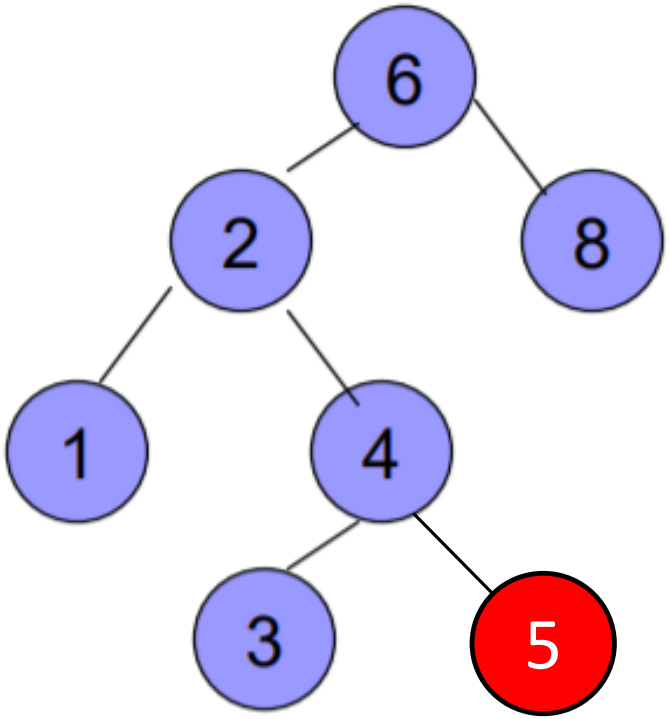
How do we insert (value) in BST's?

Let's say we want to insert **5**



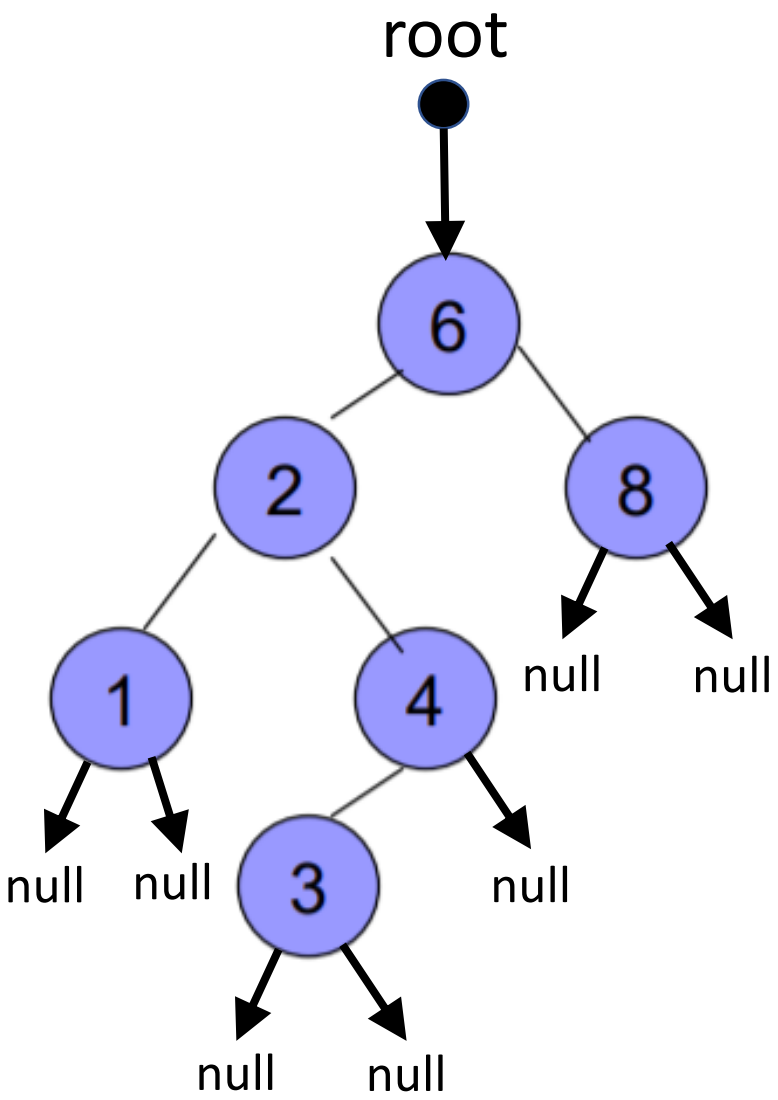
How do we insert (value) in BST's?

Let's say we want to insert **5**



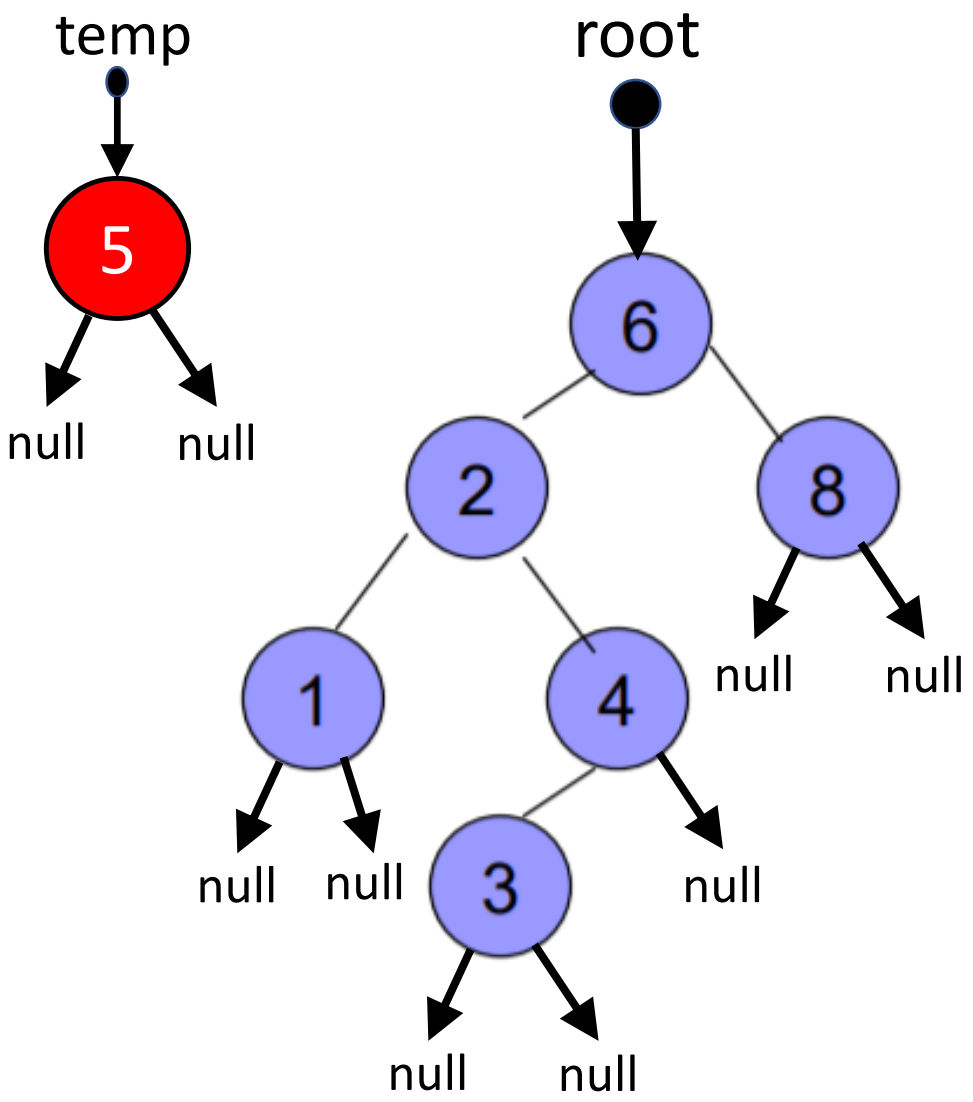
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{

}
```



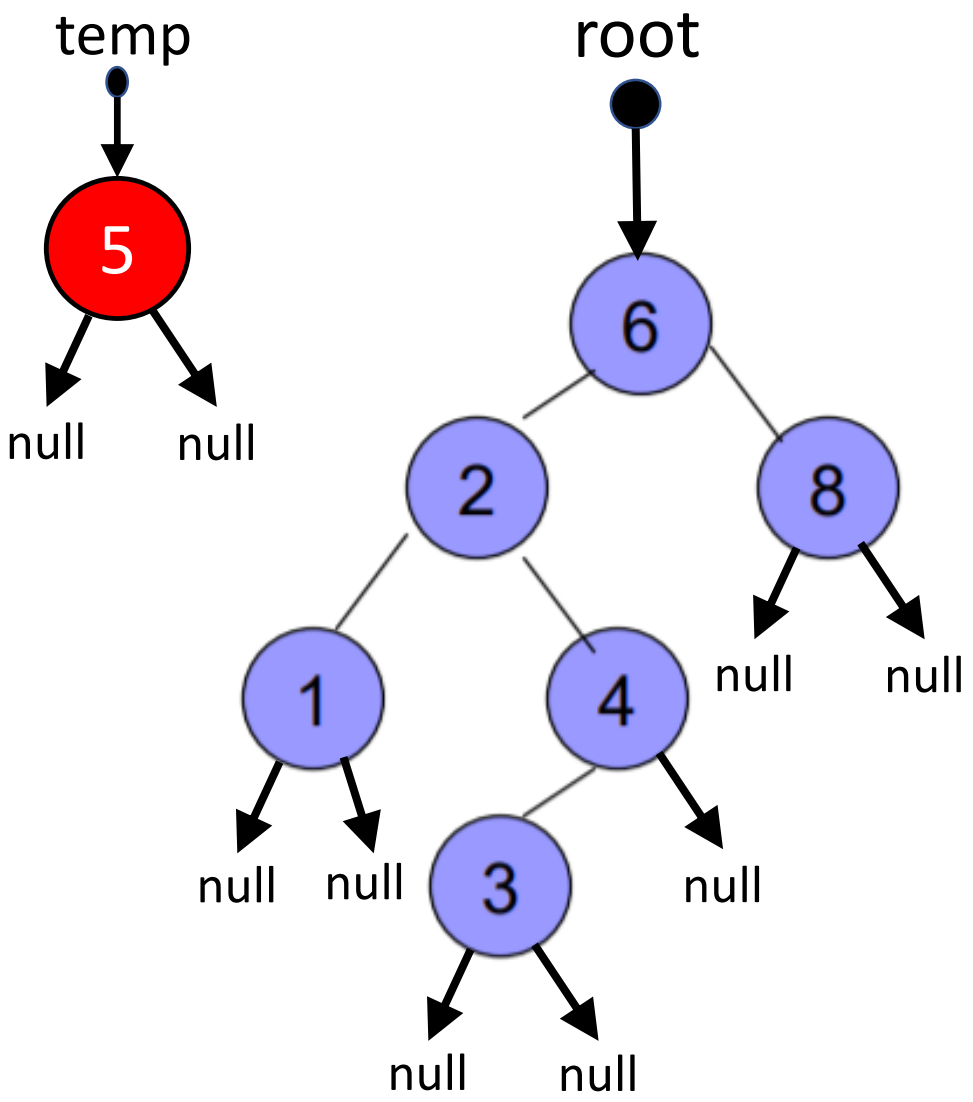
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;




}
```



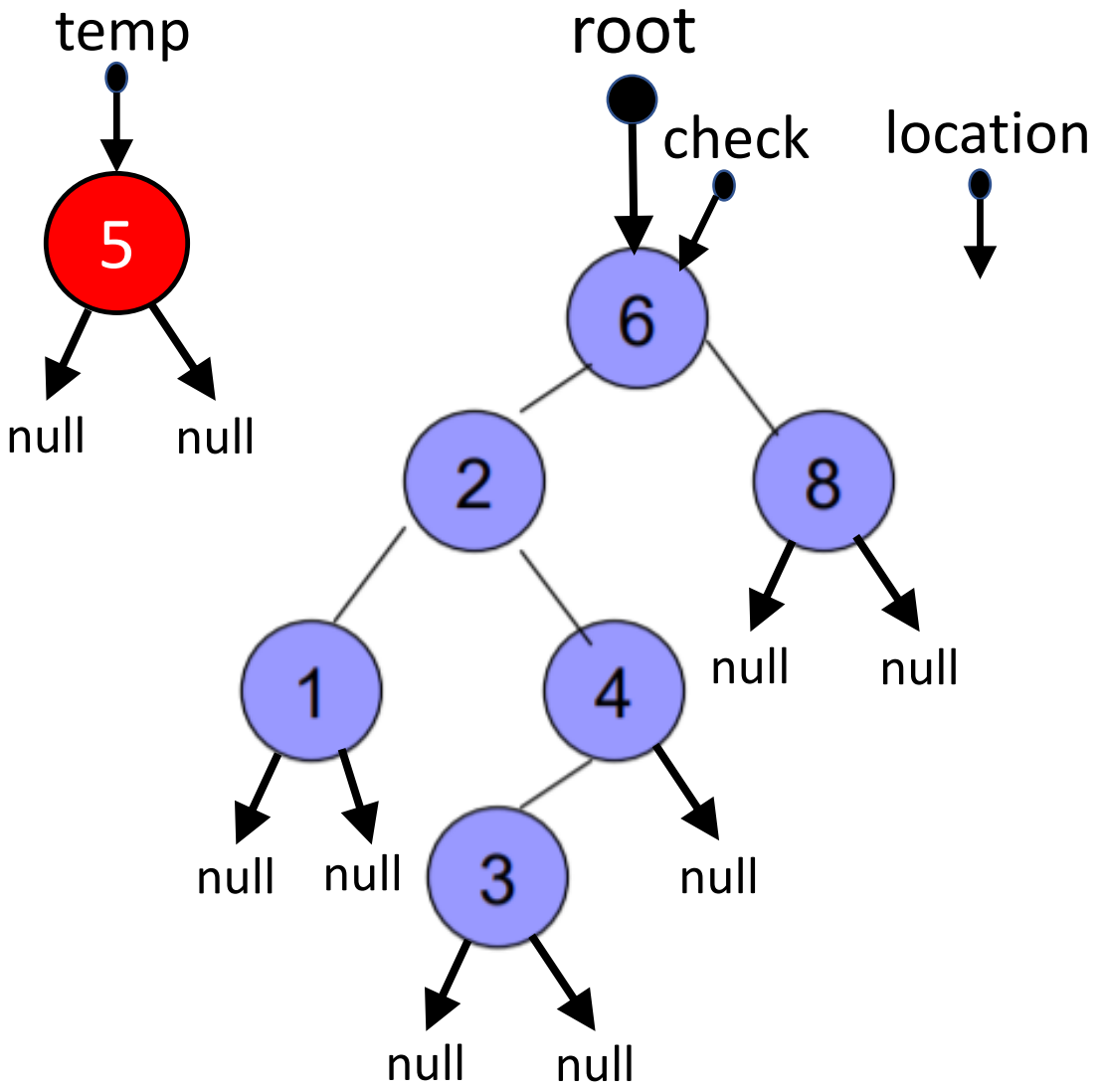
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
     Node* temp = new Node;
     temp->data = value;

     if (root == NULL){
          root = temp;
          return; }









}
```



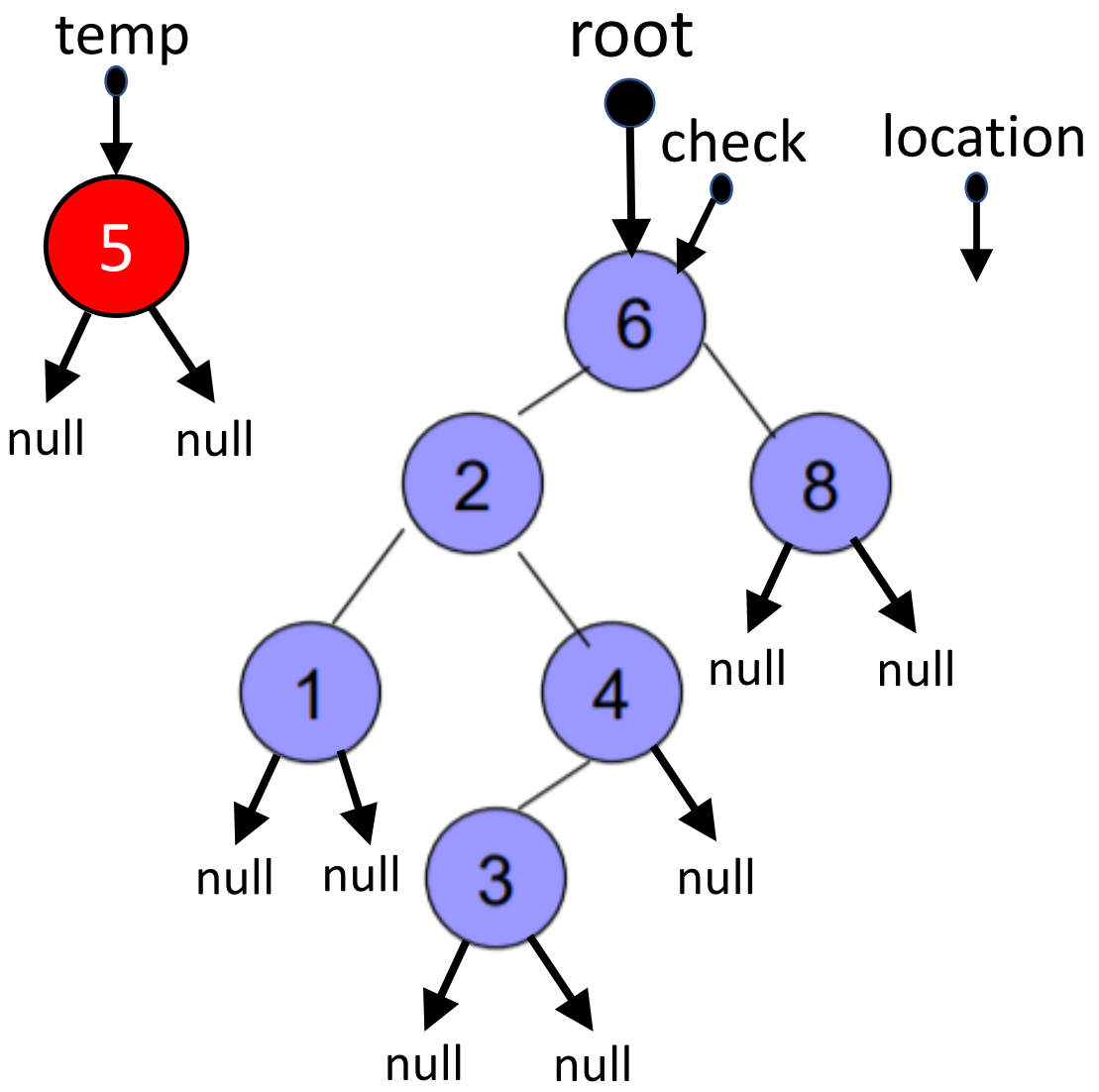
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;









}
```



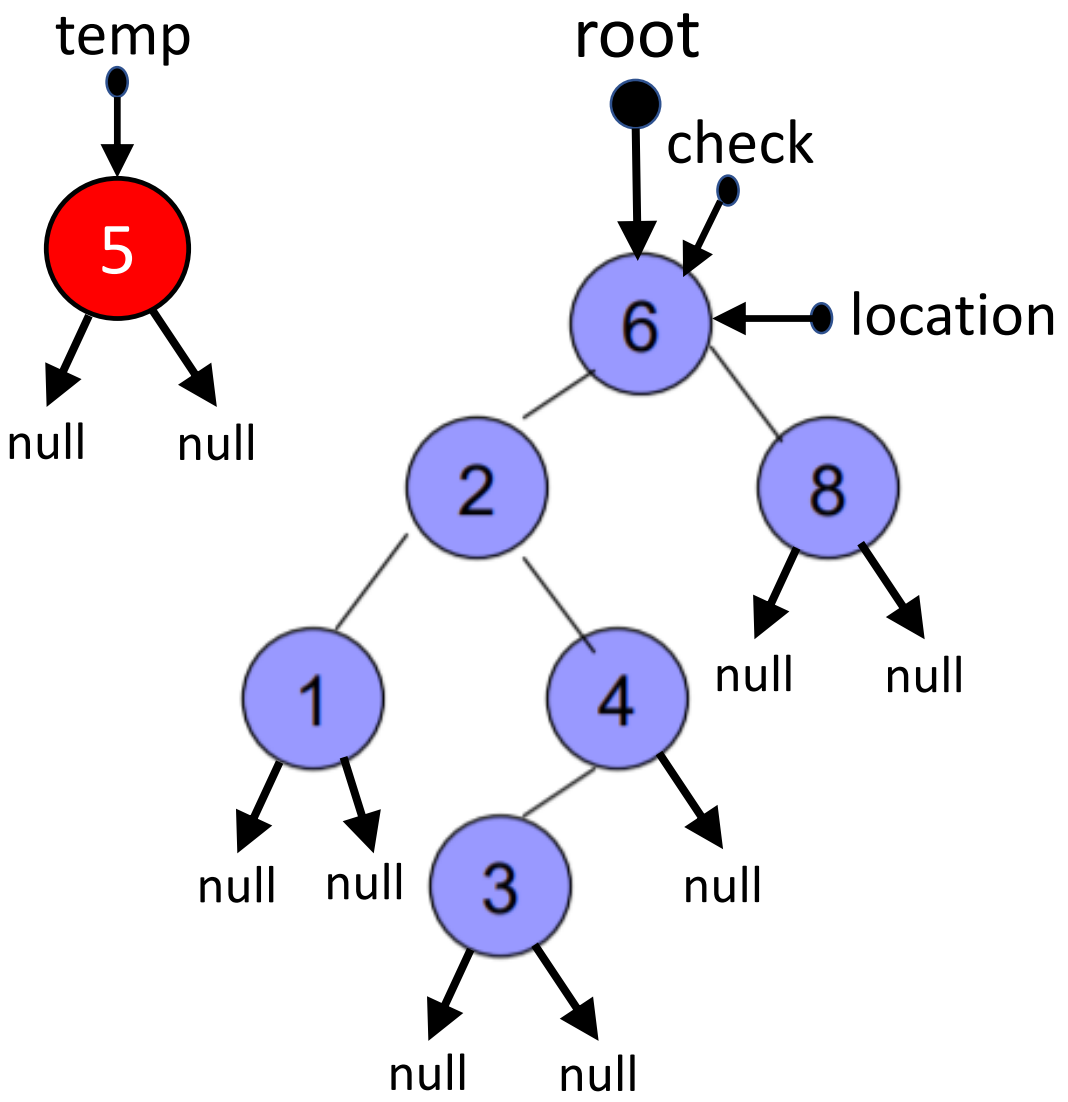
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }




}
```



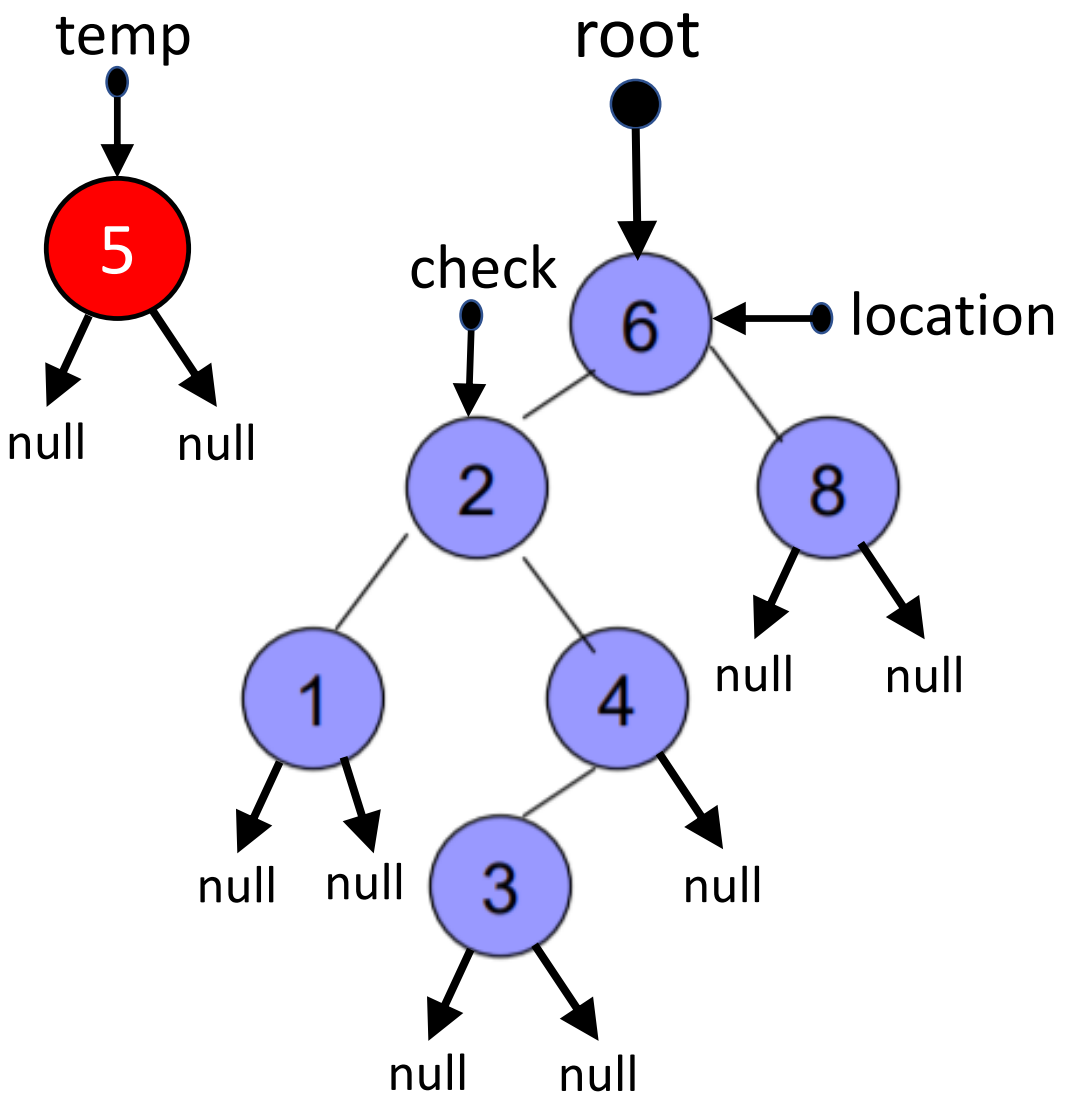
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }




}
```



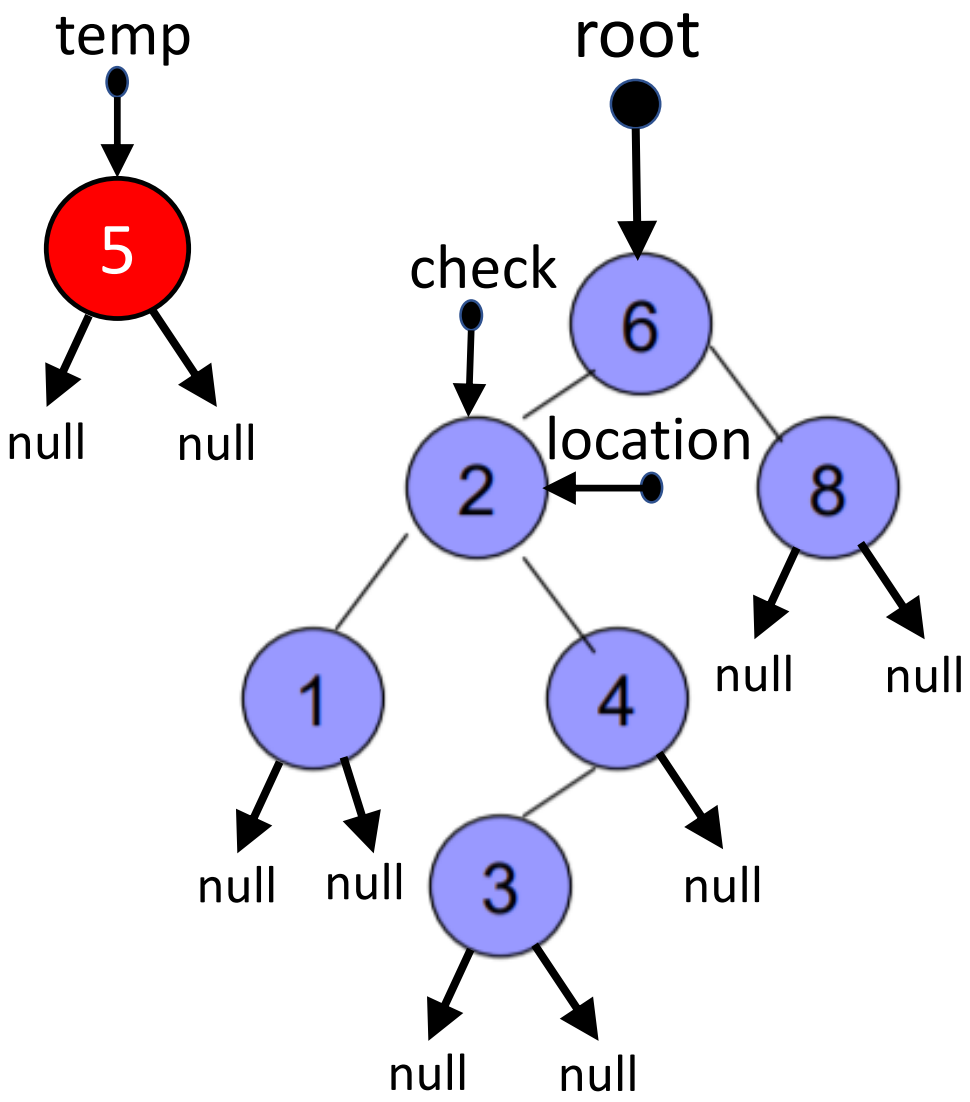
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }




}
```



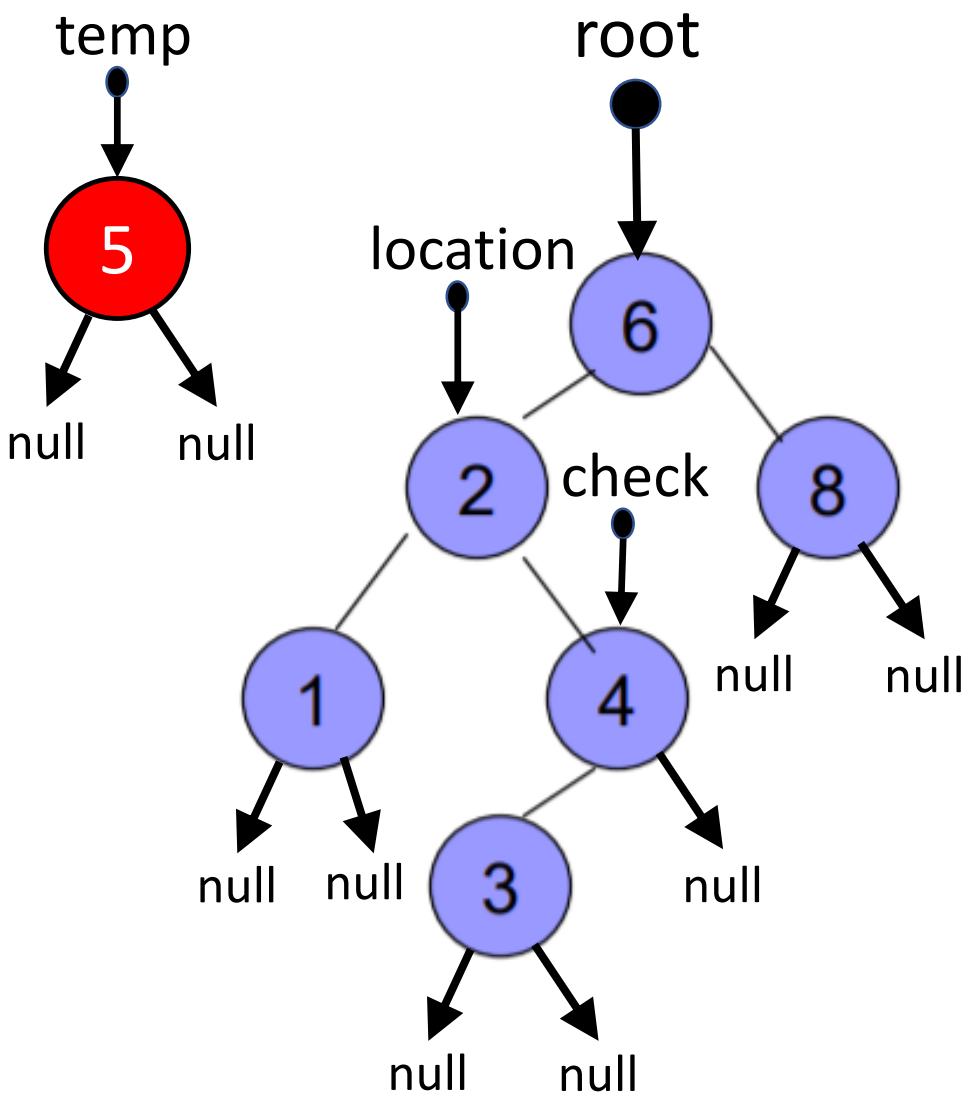
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }




}
```



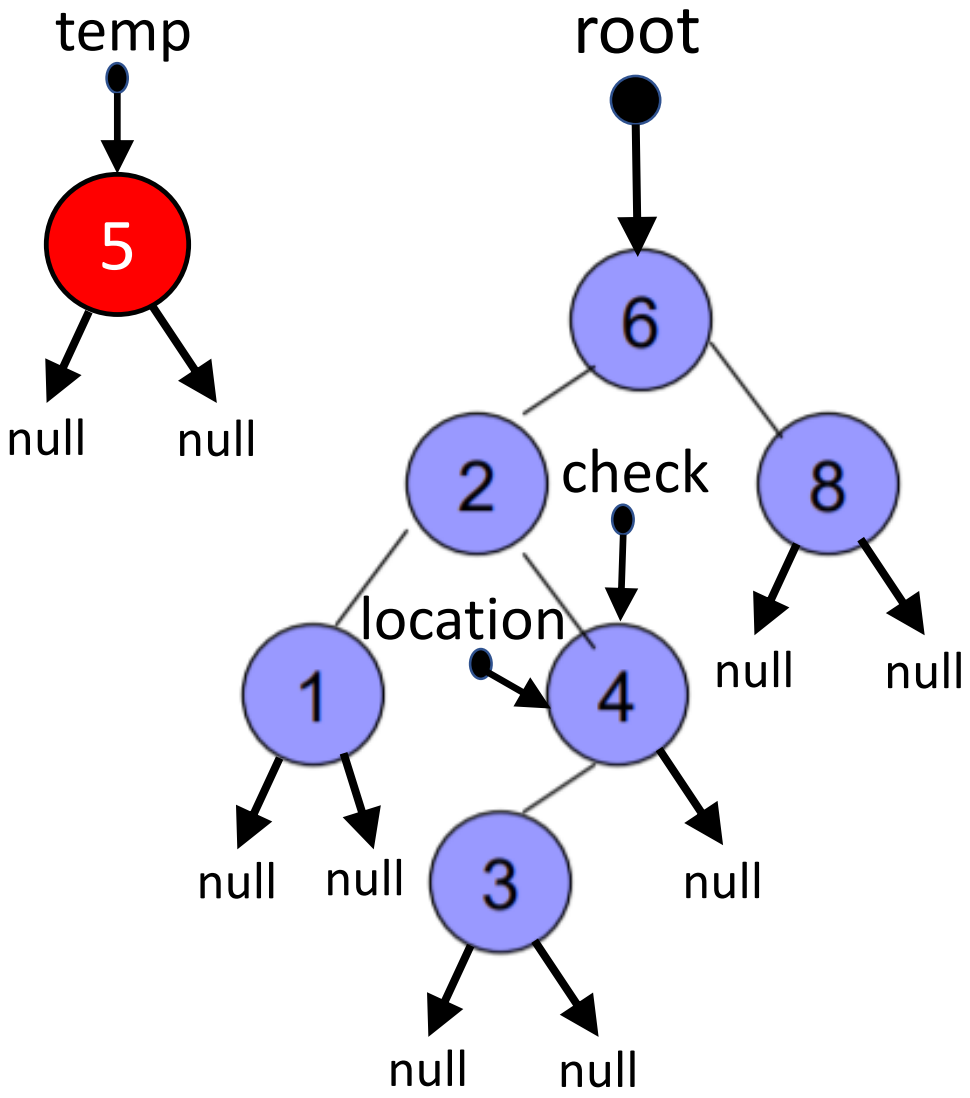
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }




}
```



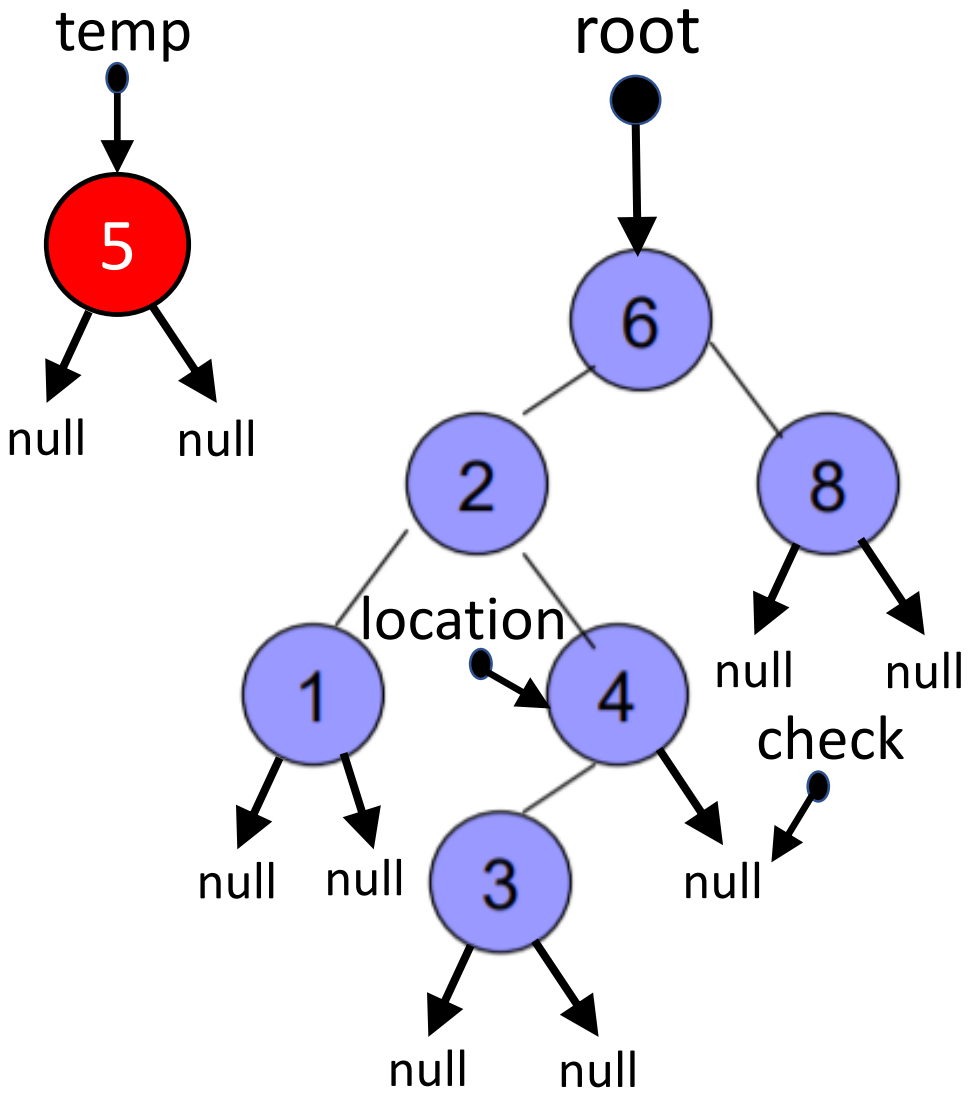
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }




}
```



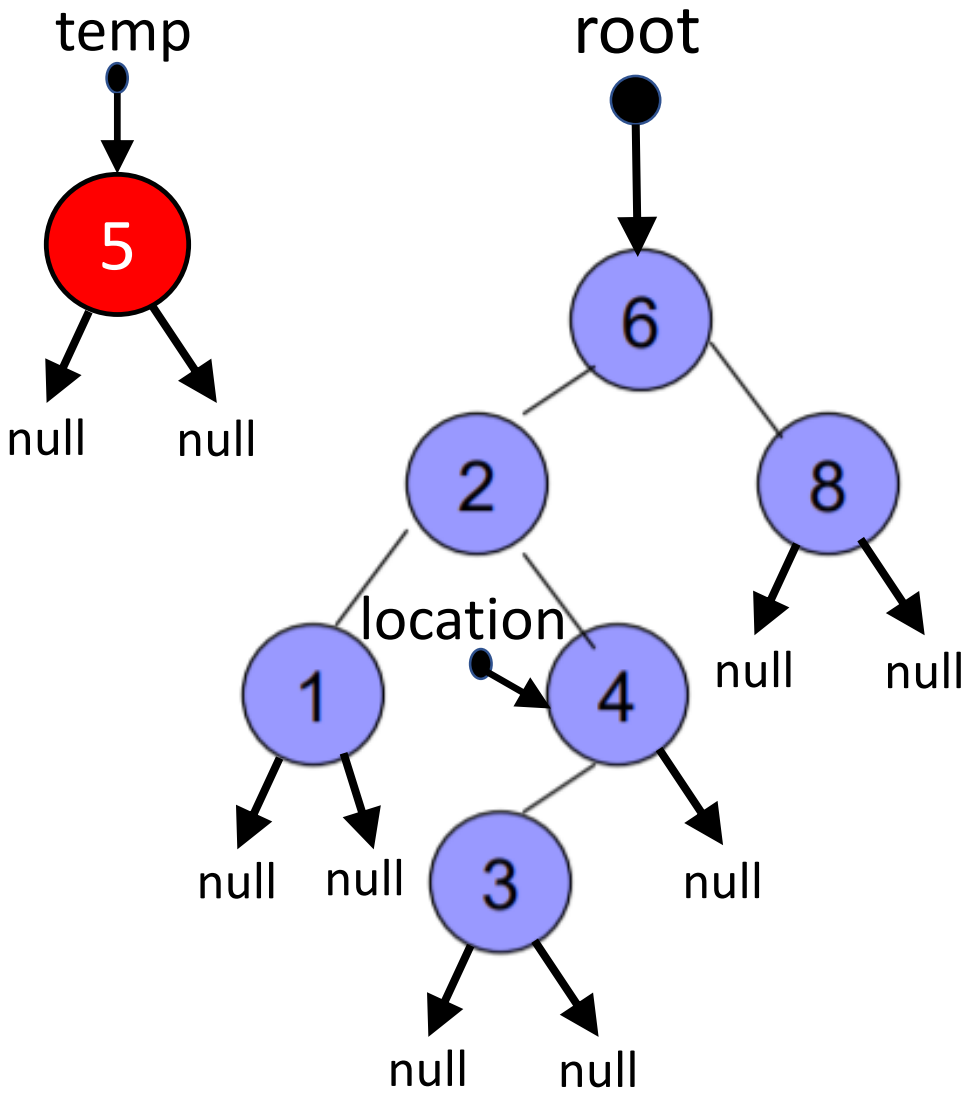
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }




}
```



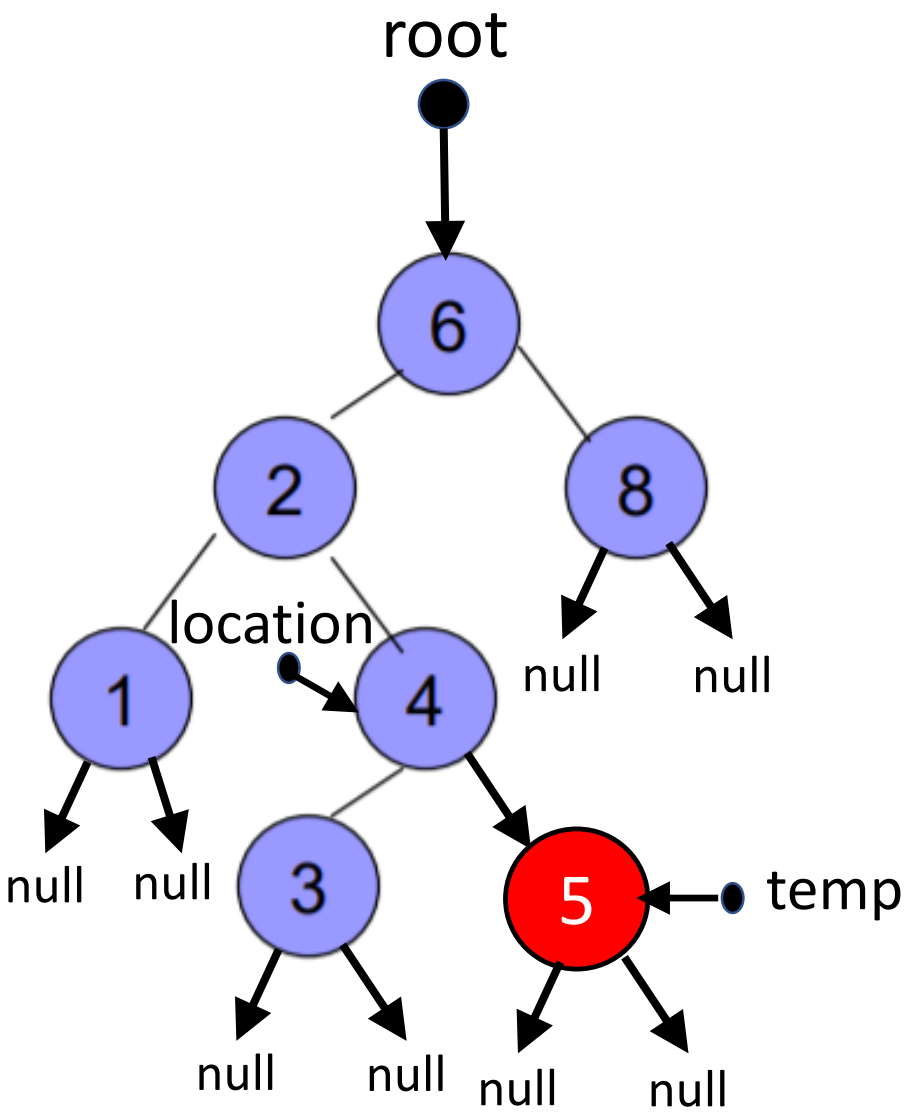
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }




}
```



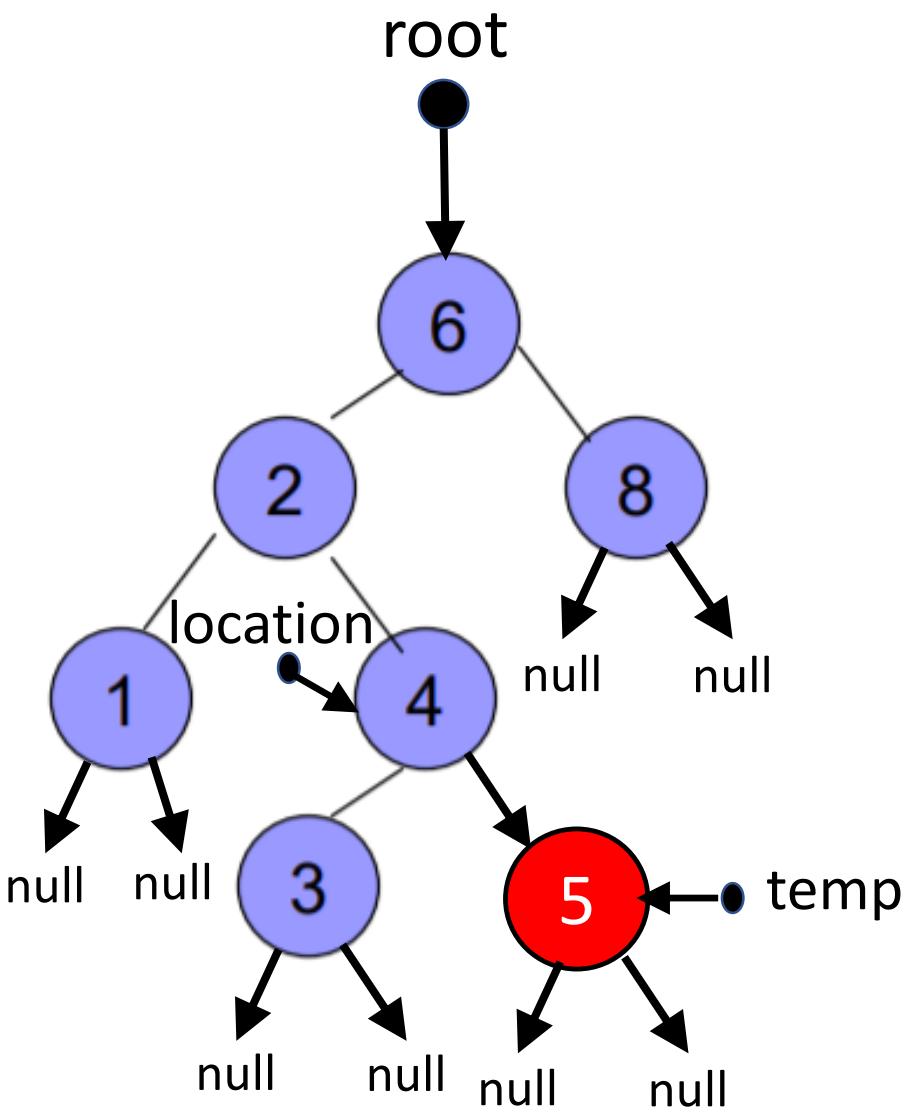
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }
    if (location->data < value)
        location->right = temp;


}
```



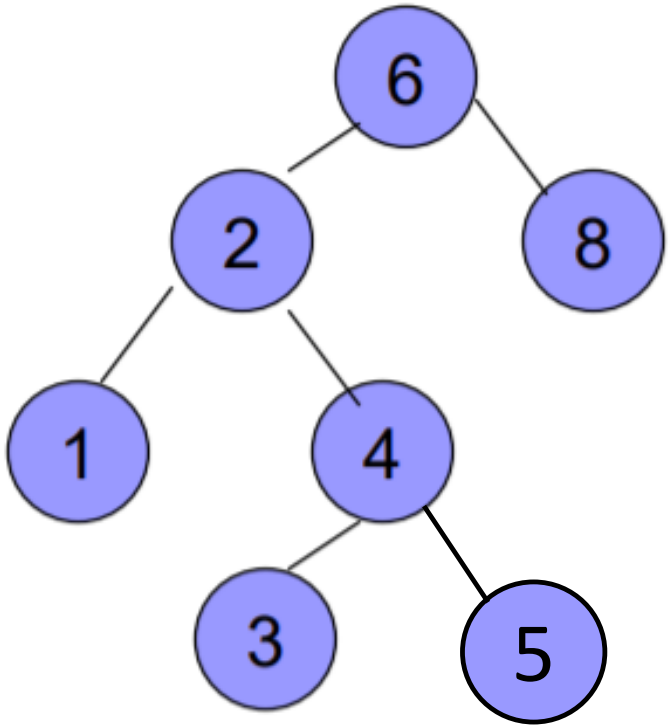
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }
    if (location->data < value)
        location->right = temp;
    else
        location->left = temp;
}
```



How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }
    if (location->data < value)
        location->right = temp;
    else
        location->left = temp;
}
```



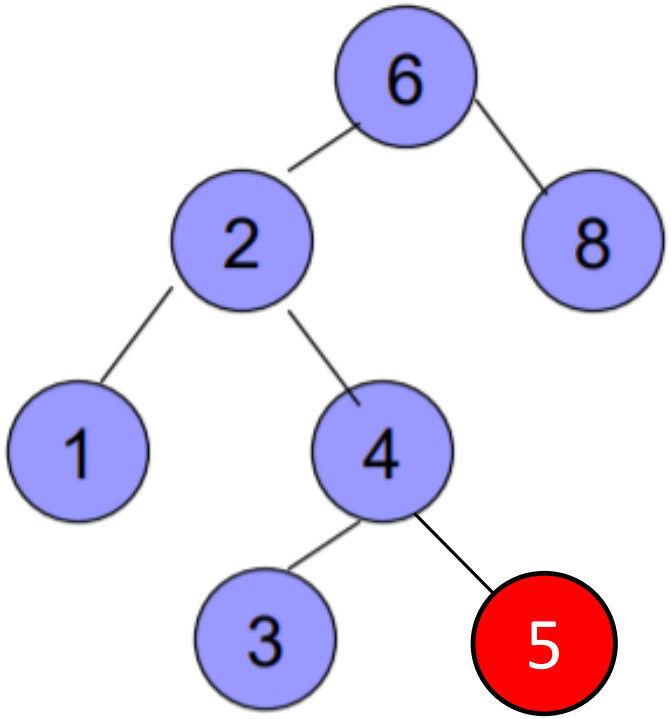
How do we insert (value) in BST's?

```
void BSTree::insert(int value)
{
    Node* temp = new Node;
    temp->data = value;

    if (root == NULL){
        root = temp;
        return; }

    Node* check = root;
    Node* location = NULL;
    while (check != NULL)
    {
        location = check;
        if (value < check->data)
            check = check->left;
        else if (value > check->data)
            check = check->right;
    }
    if (location->data < value)
        location->right = temp;
    else
        location->left = temp;
}
```

Can we do it without **location ???**

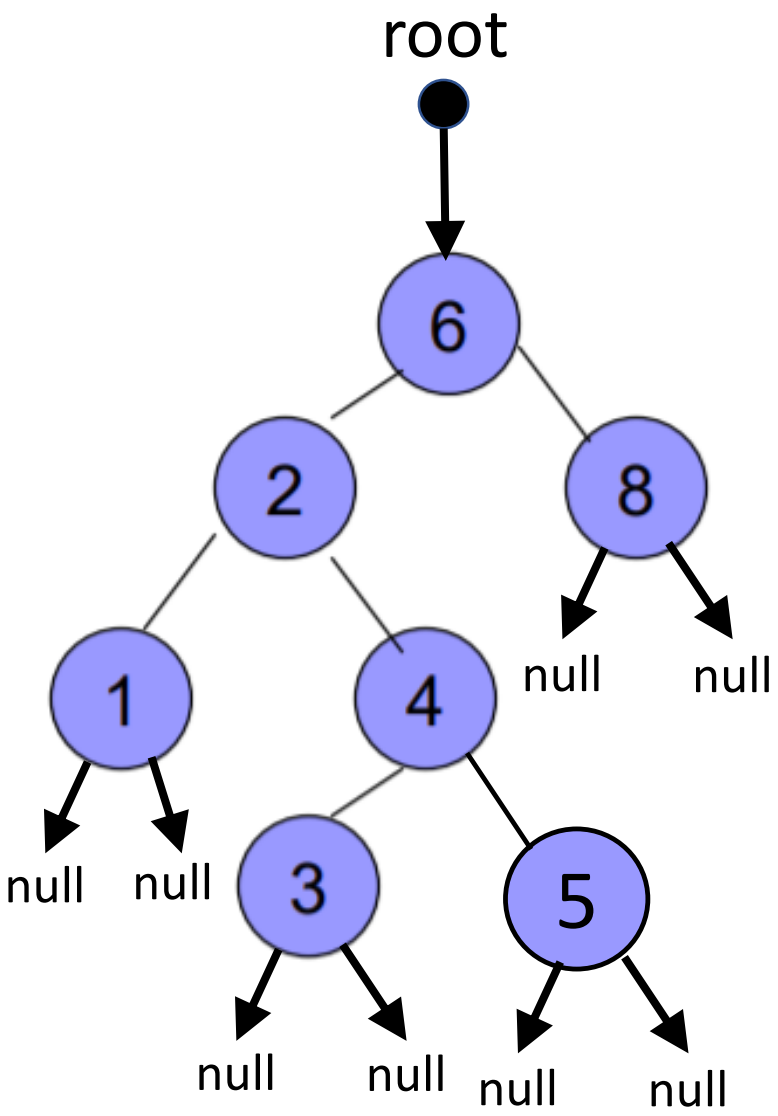
How do we find (value) in BST's?

Let's say we want to check whether **5** exists or not



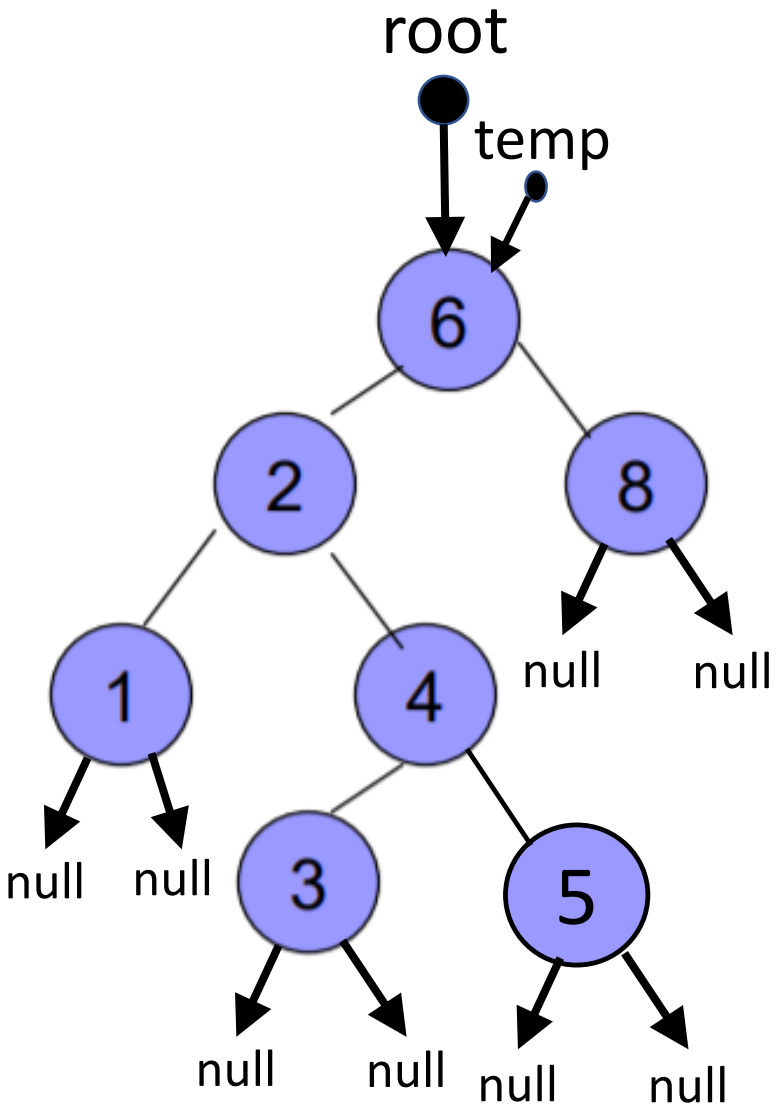
How do we find (value) in BST's?

```
bool BSTree::find(int value)
{




}
```



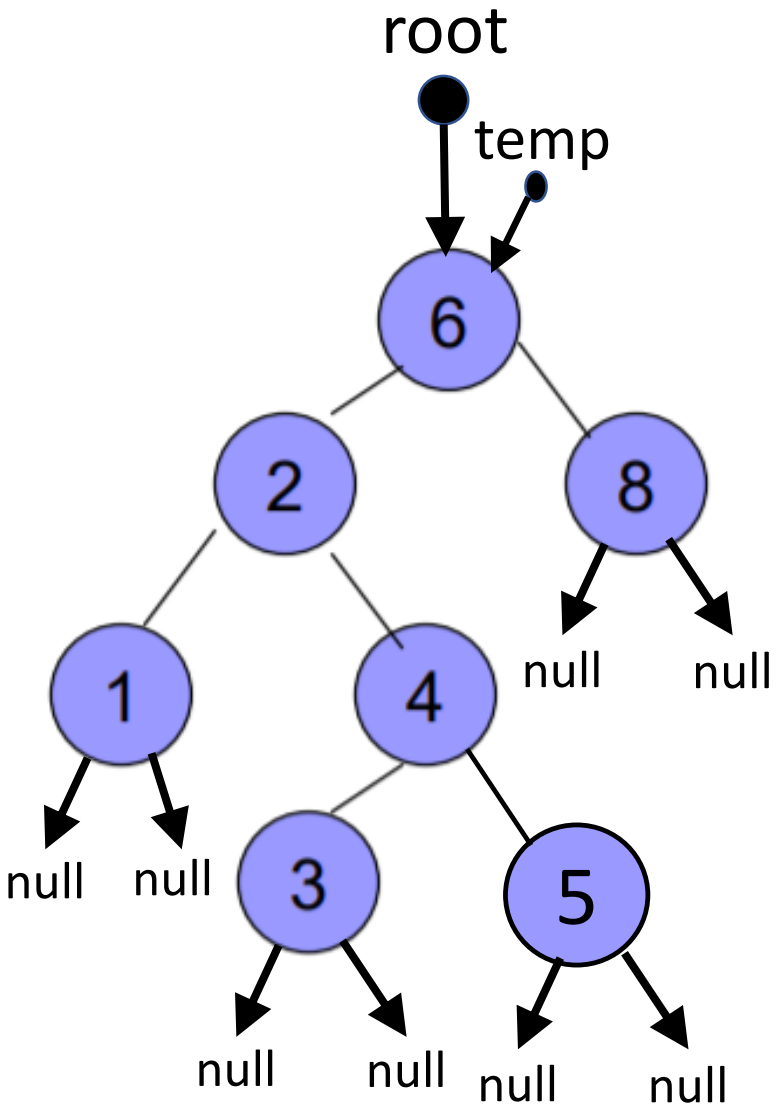
How do we find (value) in BST's?

```
bool BSTree::find(int value)
{
    Node* temp = root;




}
```



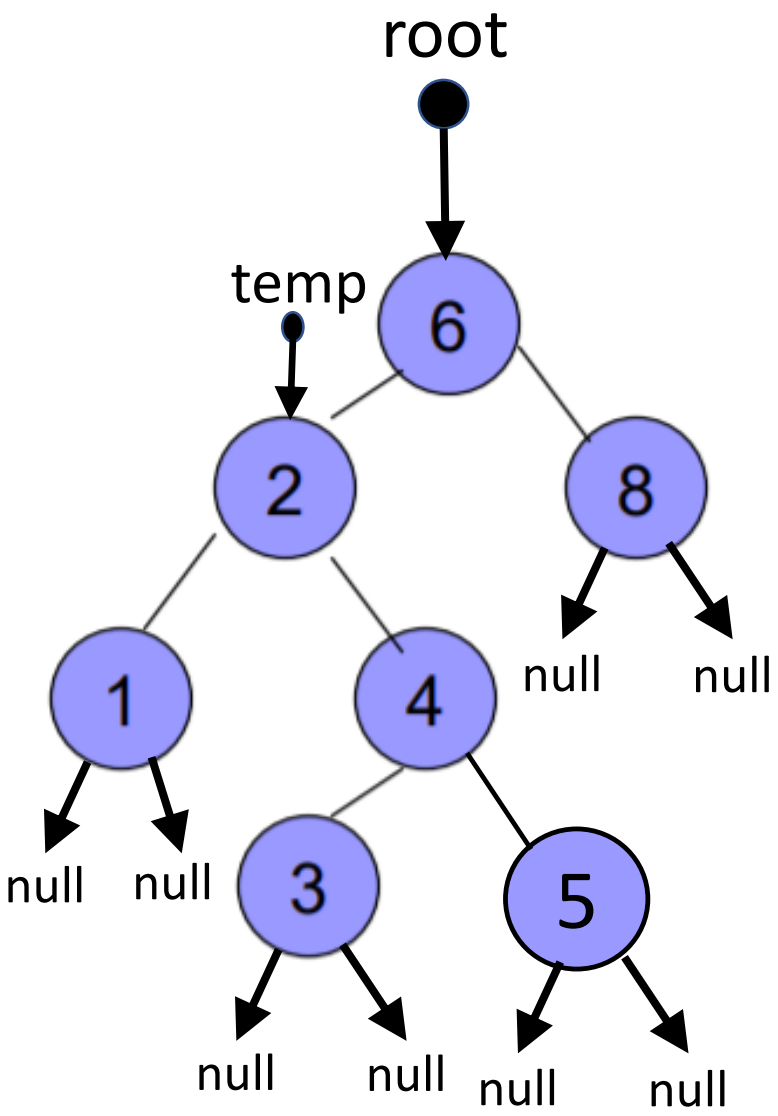
How do we find (value) in BST's?

```
bool BSTree::find(int value)
{
    Node* temp = root;

    while (temp != NULL && temp->data != value)
    {
        if (value < temp->data)
        temp = temp->left;
        else if (value > temp->data)
        temp = temp->right;
    }




}
```



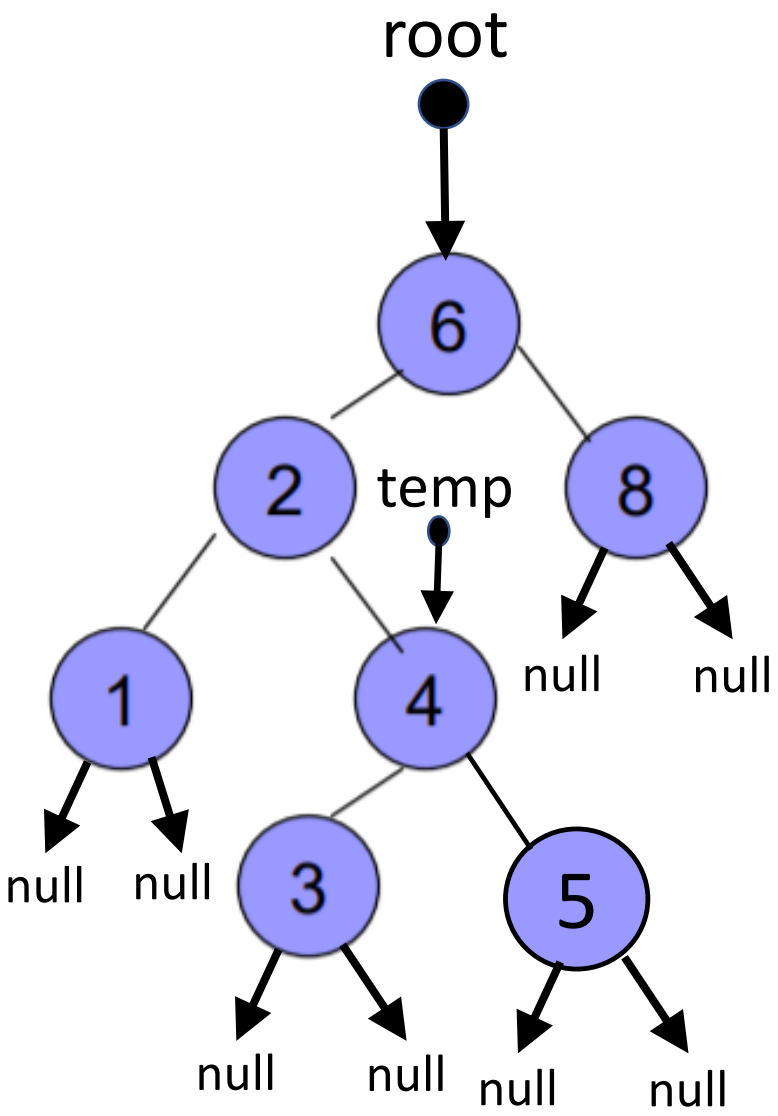
How do we find (value) in BST's?

```
bool BSTree::find(int value)
{
    Node* temp = root;

    while (temp != NULL && temp->data != value)
    {
        if (value < temp->data)
        temp = temp->left;
        else if (value > temp->data)
        temp = temp->right;
    }




}
```



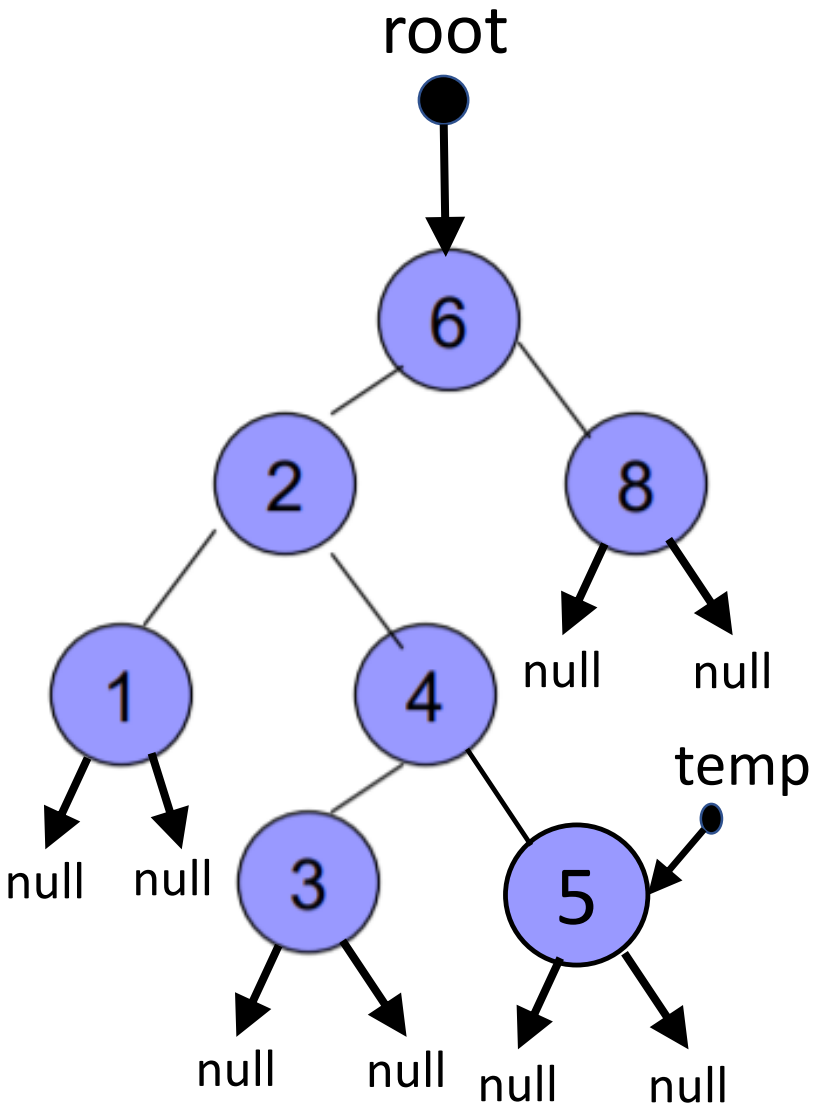
How do we find (value) in BST's?

```
bool BSTree::find(int value)
{
    Node* temp = root;

    while (temp != NULL && temp->data != value)
    {
        if (value < temp->data)
        temp = temp->left;
        else if (value > temp->data)
        temp = temp->right;
    }




}
```



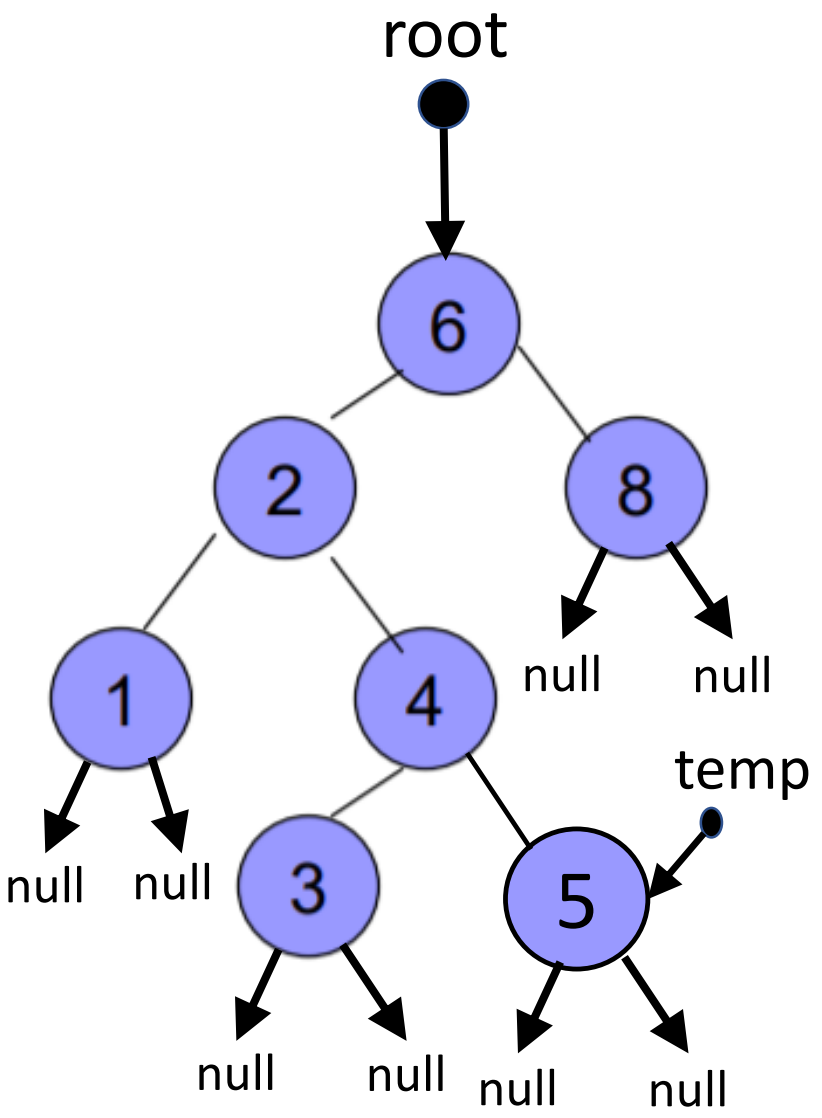
How do we find (value) in BST's?

```
bool BSTree::find(int value)
{
    Node* temp = root;

    while (temp != NULL && temp->data != value)
    {
        if (value < temp->data)
        temp = temp->left;
        else if (value > temp->data)
        temp = temp->right;
    }




}
```



How do we find (value) in BST's?

```
bool BSTree::find(int value)
{
    Node* temp = root;

    while (temp != NULL && temp->data != value)
    {
        if (value < temp->data)
        temp = temp->left;
        else if (value > temp->data)
        temp = temp->right;
    }

    if (temp == NULL)
        return 0;
    else
        return 1;
}
```



Thanks a lot

If you are taking a Nap, **wake up**.....Lecture Over