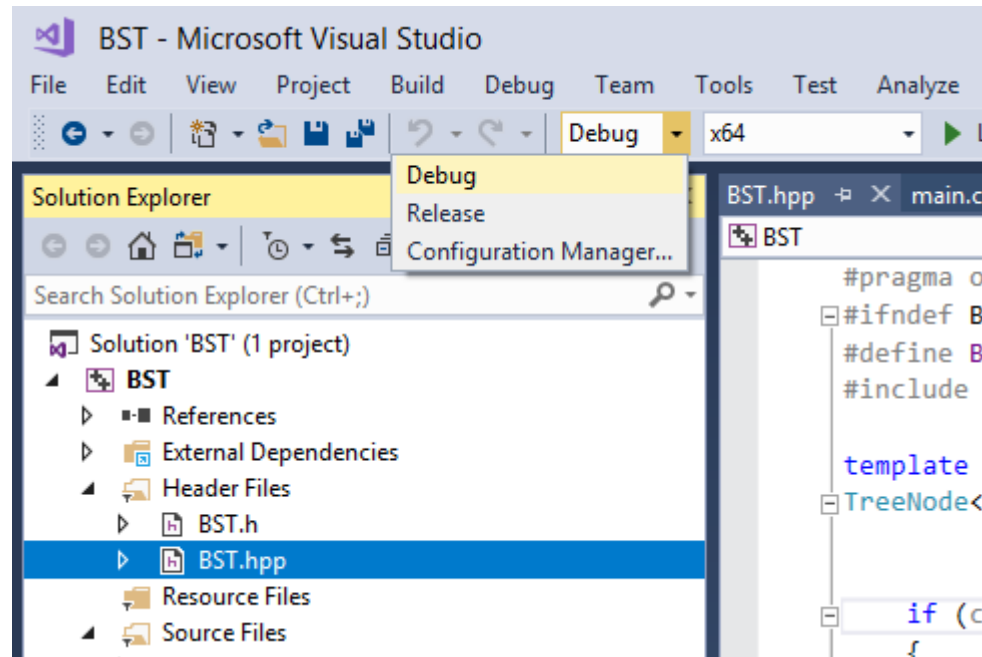


# Basic Debugging with MSVC

# Before Debugging

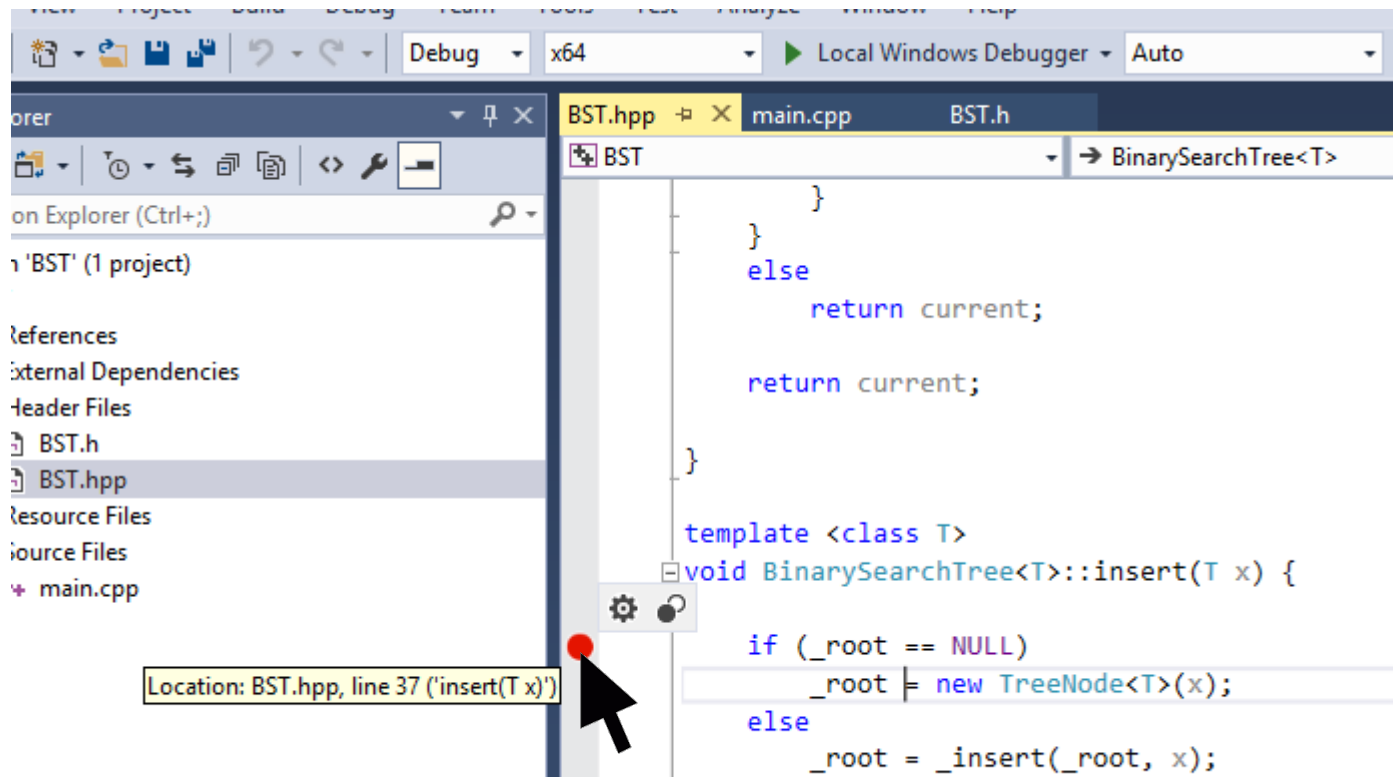
- Make sure that you compile in “Debug” profile



- Release profile will give you a faster and optimized version of executables

# Setting Breakpoints

- If you wish to pause at any line of your code, simply click a red dot on the left margin of your code



# Then Start Debugging

- By either:
  - pressing “F5”
  - or click “Local Windows Debugger”

# Debugger

You paused at this line

template <class T>  
void BinarySearchTree<T>::insert(T x) {  
 if (\_root == NULL)  
 \_root = new TreeNode<T>(x);  
 else  
 \_root = \_insert(\_root, x);  
}

**Autos**

Name	Value	Type
_root	0x0000000000000000 <NULL>	TreeNode
_item	<Unable to read memory>	
_left	<Unable to read memory>	
_right	<Unable to read memory>	
_height	<Unable to read memory>	
this	0x0000008eaa4df978 {_size=0 _root=0}	BinarySe
_size	0	int
_root	0x0000000000000000 <NULL>	TreeNode
x	7	int

**Call Stack**

Name	Lang
BST.exe!BinarySearchTree<int>::insert(int x) Line 37	C++
BST.exe!testInsertion1(bool printWithHeight) Line 27	C++
BST.exe!main() Line 19	C++
[External Code]	

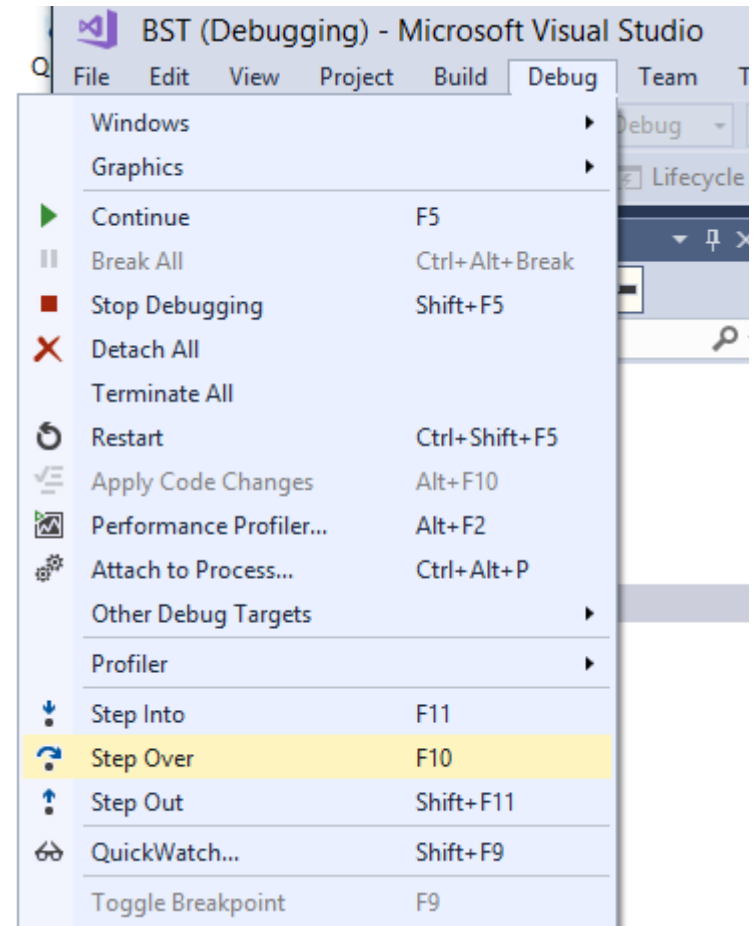
**Your call stack so far.**

- main() → testInsertion1() → insert()

All the variables at this scope

# To Continue

- You can go to the “next step” by
  - Step Over (F10)
  - Step Into (F11)
- Or simply
  - Continue (F5)
  - Run until the next breakpoint



# Let's Try "Step Over"

Press "F10" once  
Go to the next line

```
void BinarySearchTree<T>::insert(T x) {  
    if (_root == NULL)  
        _root = new TreeNode<T>(x);  
    else  
        _root = _insert(_root, x);  
}
```

Name	Value	Type
▲ _root	0x0000000000000000 <NULL>	TreeNode
✖ _item	<Unable to read memory>	
✖ _left	<Unable to read memory>	
✖ _right	<Unable to read memory>	
✖ _height	<Unable to read memory>	
▲ this	0x00000006b2e4cfb58 {_size=0 _root=0}	BinarySearchTree
_size	0	int
▲ _root	0x0000000000000000 <NULL>	TreeNode
✖ _item	<Unable to read memory>	
✖ _left	<Unable to read memory>	
✖ _right	<Unable to read memory>	
✖ _height	<Unable to read memory>	
x	7	int

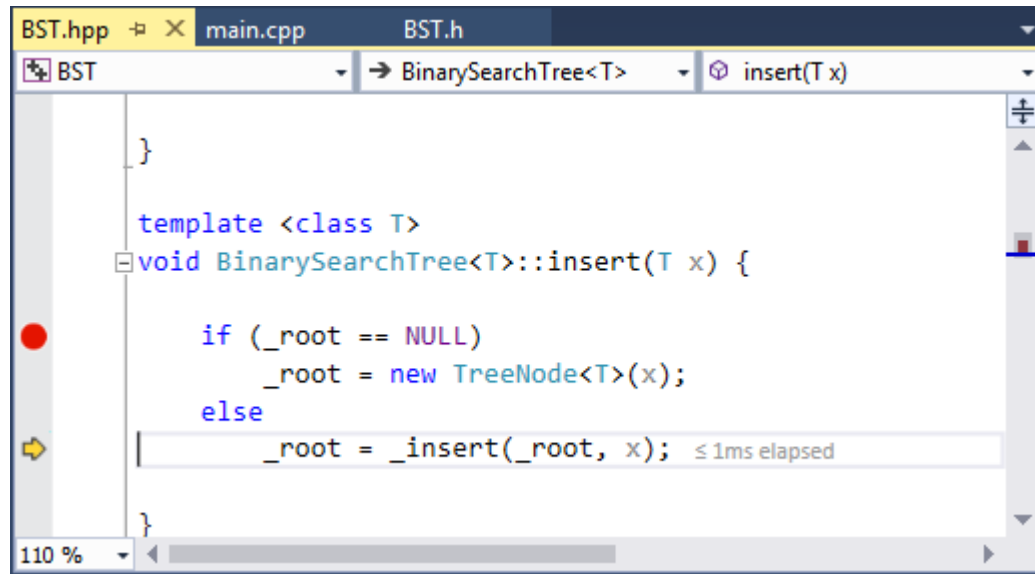
Press "F10" again  
Go to the next line

```
void BinarySearchTree<T>::insert(T x) {  
    if (_root == NULL)  
        _root = new TreeNode<T>(x);  
    else  
        _root = _insert(_root, x);  
}
```

Name	Value	Type
operator new	0x00000006b2e5388c0	void *
▲ _root	0x00000006b2e5388c0 {_item=7 _left=0}	TreeNode
_size	7	int
▲ _left	0x0000000000000000 <NULL>	TreeNode
▲ _right	0x0000000000000000 <NULL>	TreeNode
_size	0	int
▲ this	0x00000006b2e4cfb58 {_size=0 _root=0}	BinarySearchTree
_size	0	int
▲ _root	0x00000006b2e5388c0 {_item=7 _left=0}	TreeNode
_size	7	int
▲ _left	0x0000000000000000 <NULL>	TreeNode
▲ _right	0x0000000000000000 <NULL>	TreeNode
_size	0	int
x	7	int

Notice all the changes  
in the variables here  
(marked red)

- At this point, you have inserted a node 7
- Press “F5” to continue to the next breakpoint
- After one “F10”, you will be here:



The screenshot shows a debugger window with three tabs: BST.hpp, main.cpp, and BST.h. The BST.h tab is active, showing the `BinarySearchTree<T>` class. The `insert(T x)` function is selected in the right-hand pane. The left-hand pane shows a red dot indicating a breakpoint at the start of the `insert` function. The main window displays the following code:

```
template <class T>
void BinarySearchTree<T>::insert(T x) {
    if (_root == NULL)
        _root = new TreeNode<T>(x);
    else
        _root = _insert(_root, x);
}
```

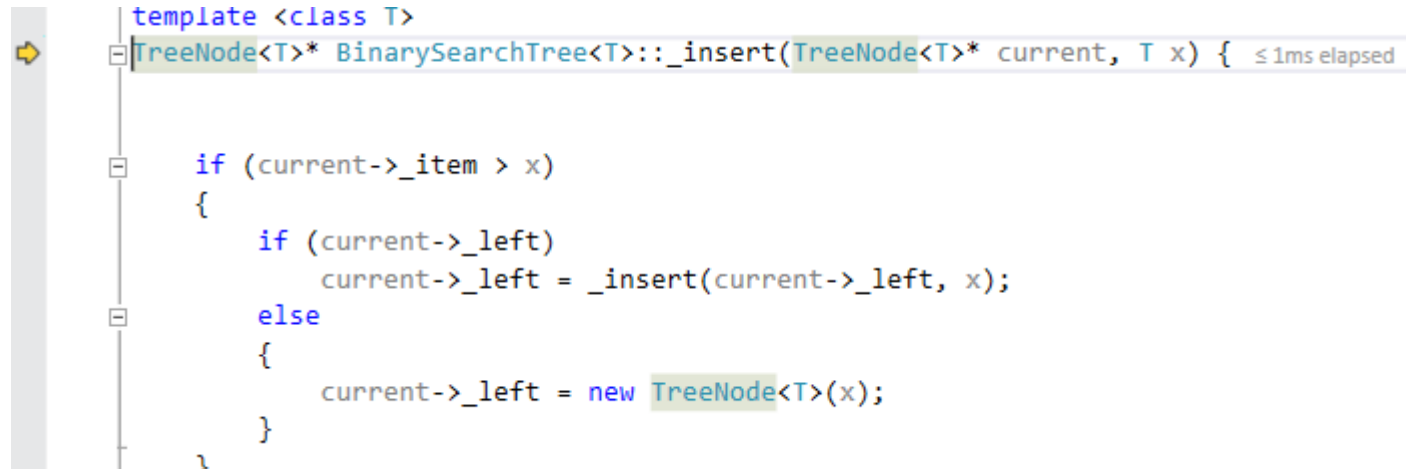
A yellow arrow points to the line `_root = _insert(_root, x);`, which has a tooltip indicating `≤ 1ms elapsed`. The status bar at the bottom shows `110 %`.

- Because you have inserted a node and `_root` is not NULL now
- Now you have a choice, “F10” or “F11”



- F11

- Will go into the function “\_insert()”



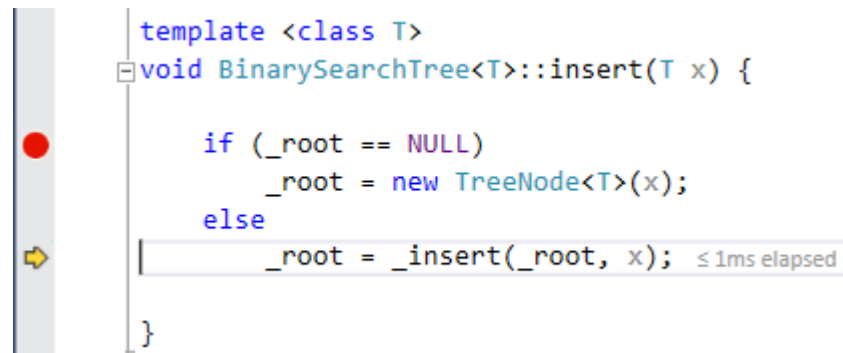
A screenshot of a debugger window showing the source code of the `_insert` function in the `BinarySearchTree` class. The function is a template method that takes a `TreeNode` pointer and a value `x`. The code is as follows:

```
template <class T>
TreeNode<T>* BinarySearchTree<T>::_insert(TreeNode<T>* current, T x) { ≤ 1ms elapsed

    if (current->_item > x)
    {
        if (current->_left)
            current->_left = _insert(current->_left, x);
        else
        {
            current->_left = new TreeNode<T>(x);
        }
    }
}
```

The debugger interface includes a vertical timeline on the left with a yellow arrow pointing to the first line of the function. The code is color-coded: keywords in blue, identifiers in black, and literals in red.

- But if you press F10 instead, it will “finish” the function `_insert()`



A screenshot of a debugger window showing the source code of the `insert` function in the `BinarySearchTree` class. The function is a template method that takes a value `x`. The code is as follows:

```
template <class T>
void BinarySearchTree<T>::insert(T x) {

    if (_root == NULL)
        _root = new TreeNode<T>(x);
    else
        _root = _insert(_root, x); ≤ 1ms elapsed
}
```

The debugger interface includes a vertical timeline on the left with a red dot at the top and a yellow arrow pointing to the last line of the function. The code is color-coded: keywords in blue, identifiers in black, and literals in red.

# After Finishing \_insert()

- \_left of the \_root is not NULL anymore

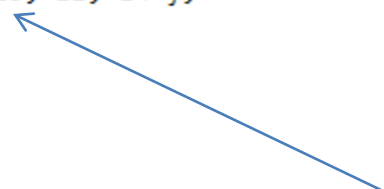
Autos		
Name	Value	
BinarySearchTree<int>::_insert return	0x0000006b2e5388c0 {_item=7 _left=0x0000006b2e538920 _right=0x0000000000000000 ..	T
▸ _root	0x0000006b2e5388c0 {_item=7 _left=0x0000006b2e538920 {_item=3 _left=0x0000000000000000	T
▸ this	0x0000006b2e4cfb58 {_size=0 _root=0x0000006b2e5388c0 {_item=7 _left=0x0000006b2e538920	E
▸ _size	0	ii
▸ _root	0x0000006b2e5388c0 {_item=7 _left=0x0000006b2e538920 {_item=3 _left=0x0000000000000000	T
▸ _item	7	ii
▸ _left	0x0000006b2e538920 {_item=3 _left=0x0000000000000000 <NULL> _right=0x0000000000000000	T
▸ _right	0x0000000000000000 <NULL>	T
▸ _height	0	ii
▸ x	3	ii

Autos			
Name	Value	Type	
BinarySearchTree<int>::_insert return	0x0000006b2e5388c0 {_item=7 _left=0x0000006b2e538920 _right=0x0000000000000000 ..	TreeNode	
▸ _root	0x0000006b2e5388c0 {_item=7 _left=0x0000006b2e538920 {_item=3 _left=0x0000000000000000	TreeNode	
▸ this	0x0000006b2e4cfb58 {_size=0 _root=0x0000006b2e5388c0 {_item=7 _left=0x0000006b2e538920	BinarySe	
▸ _size	0	int	
▸ _root	0x0000006b2e5388c0 {_item=7 _left=0x0000006b2e538920 {_item=3 _left=0x0000000000000000	TreeNode	
▸ _item	7	int	
▸ _left	0x0000006b2e538920 {_item=3 _left=0x0000000000000000 <NULL> _right=0x0000000000000000	TreeNode	
▸ _item	3	int	
▸ _left	0x0000000000000000 <NULL>	TreeNode	
▸ _right	0x0000000000000000 <NULL>	TreeNode	
▸ _height	0	int	
▸ _right	0x0000000000000000 <NULL>	TreeNode	

# Conditional Breakpoint

- The code insert a lot of numbers

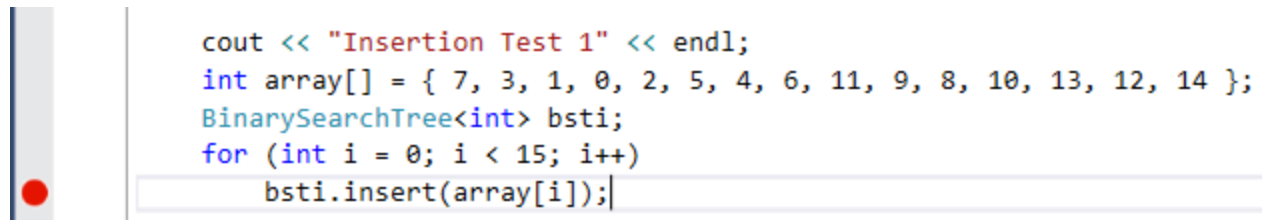
```
cout << "Insertion Test 1" << endl;  
int array[] = { 7, 3, 1, 0, 2, 5, 4, 6, 11, 9, 8, 10, 13, 12, 14 };  
BinarySearchTree<int> bsti;  
for (int i = 0; i < 15; i++)  
    bsti.insert(array[i]);
```



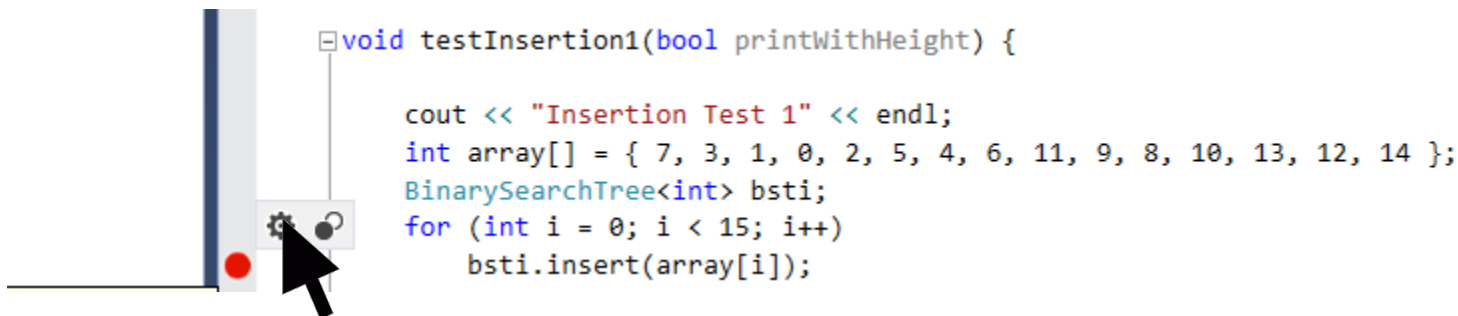
- If there is something wrong when I insert 13, I have to
  - F10, F11, F11, F10, F11, F10, F11, .... ?

# Conditional Breakpoint

- Set a breakpoint here



- Click “setting”



# Setting Condition to Break

- Then you can check “conditions”
  - Add “array[i]==13”
  - Then “close”

```
BinarySearchTreeNode bst1;  
for (int i = 0; i < 15; i++)  
    bst1.insert(array[i]);
```

Location: main.cpp, Line: 27, Must match source

☒ Conditions

Conditional Expression

Is true

array[i]==13

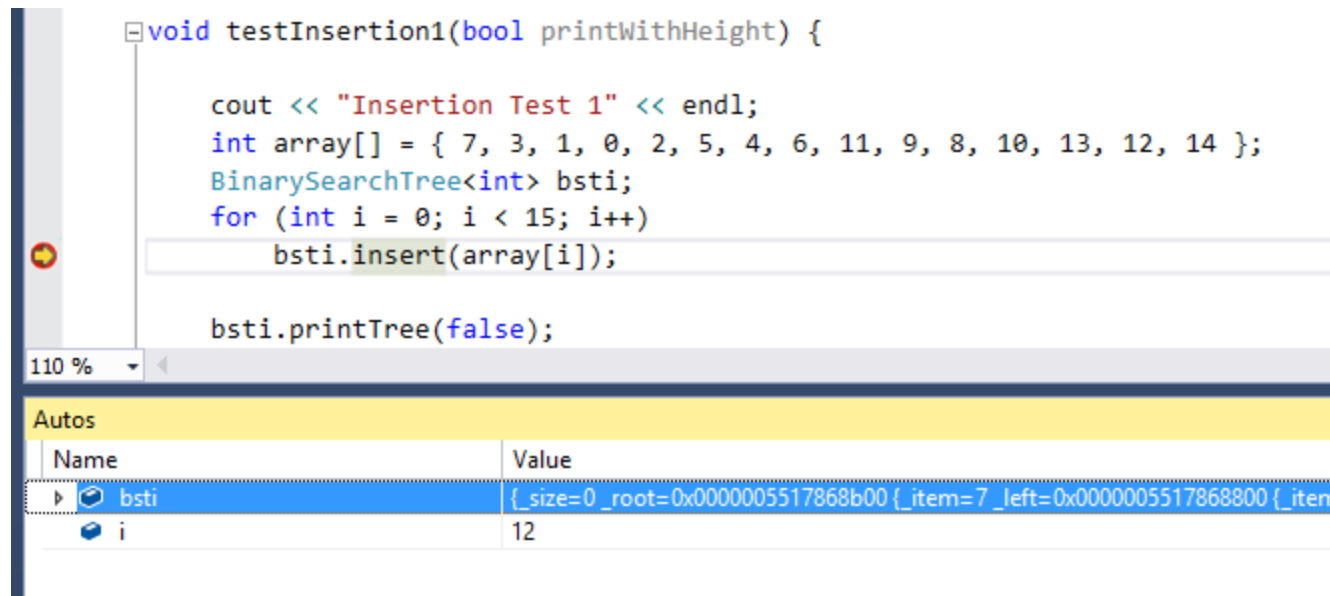
Add condition

☐ Actions

Close

# When You Run

- The program will pause at `array[i] == 13`
  - Unless you haven't remove the previous breakpoint



```
void testInsertion1(bool printWithHeight) {  
  
    cout << "Insertion Test 1" << endl;  
    int array[] = { 7, 3, 1, 0, 2, 5, 4, 6, 11, 9, 8, 10, 13, 12, 14 };  
    BinarySearchTreeNode bsti;  
    for (int i = 0; i < 15; i++)  
        bsti.insert(array[i]);  
  
    bsti.printTree(false);  
}
```

110 %

Autos	
Name	Value
bsti	{_size=0 _root=0x0000005517868b00 {_item=7 _left=0x0000005517868800 {_item=7 _right=0x0000005517868800}}
i	12