

```

1  #include "BSTree.h"
2
3  BSTree::BSTree()
4  : pObjBSTNodeRoot( NULL ), uintSize( 0 )
5  {
6      /* empty */
7  }
8
9  BSTree::~~BSTree()
10 {
11     this->clear();
12 }
13
14 bool BSTree::insert( int intInsertValue )
15 {
16     this->insert( intInsertValue, this->pObjBSTNodeRoot );
17 }
18
19 void BSTree::clear()
20 {
21     this->clear( this->pObjBSTNodeRoot );
22 }
23
24 unsigned int BSTree::getSize() const
25 {
26     return this->uintSize;
27 }
28
29 void BSTree::inOrder()
30 {
31     this->inOrder( this->pObjBSTNodeRoot );
32 }
33
34 bool BSTree::insert( int intInsertValue, BSTNode*& pObjBSTNodeInsertNode )
35 {
36     if( uintSize > 0 )
37     {
38         if( intInsertValue == pObjBSTNodeInsertNode->getContents() )
39         {
40             return false;
41         }
42         else
43         {
44             if( intInsertValue < pObjBSTNodeInsertNode->getContents() )
45             {
46                 if( pObjBSTNodeInsertNode->getLeftChild() == NULL )
47                 {
48                     pObjBSTNodeInsertNode->setLeftChild( new BSTNode( intInsertValue ) );
49                     this->uintSize++;
50                     return true;

```

```

51         }
52         else
53         {
54             return this->insert( intInsertValue, pObjBSTNodeInsertNode->getLeftChild() );
55         }
56     }
57
58     if( intInsertValue > pObjBSTNodeInsertNode->getContents() )
59     {
60         if( pObjBSTNodeInsertNode->getRightChild() == NULL )
61         {
62             pObjBSTNodeInsertNode->setRightChild( new BSTNode( intInsertValue ) );
63             this->uintSize++;
64             return true;
65         }
66         else
67         {
68             return this->insert( intInsertValue, pObjBSTNodeInsertNode->getRightChild() );
69         }
70     }
71 }
72 }
73 else
74 {
75     pObjBSTNodeInsertNode = new BSTNode( intInsertValue );
76     this->uintSize++;
77     return true;
78 }
79 }
80
81 void BSTree::clear( BSTNode*& pObjBSTNodeClearNode )
82 {
83     if( uintSize > 0 )
84     {
85         if( pObjBSTNodeClearNode->getLeftChild() != NULL )
86         {
87             this->clear( pObjBSTNodeClearNode->getLeftChild() );
88         }
89
90         if( pObjBSTNodeClearNode->getRightChild() != NULL )
91         {
92             this->clear( pObjBSTNodeClearNode->getRightChild() );
93         }
94
95         delete pObjBSTNodeClearNode;
96         pObjBSTNodeClearNode = NULL;
97         this->uintSize--;
98     }
99 }
100

```

```
101 void BSTree::inOrder( BSTNode* pObjBSTNodePrintNode )
102 {
103     if( uintSize > 0 )
104     {
105         if( pObjBSTNodePrintNode->getLeftChild() != NULL )
106         {
107             this->inOrder( pObjBSTNodePrintNode->getLeftChild() );
108         }
109
110         cout << pObjBSTNodePrintNode->getContents() << " ";
111
112         if( pObjBSTNodePrintNode->getRightChild() != NULL )
113         {
114             this->inOrder( pObjBSTNodePrintNode->getRightChild() );
115         }
116     }
117     else
118     {
119         cout << "";
120     }
121 }
122
```