Binary Search Trees

In this lab, you are going to implement a binary search tree class of integers using arrays. This may seem confusing at first but this is possible because we know the arrangement of the nodes in the tree follows a pattern. If we consider the root of tree to be at index 0, we can calculate the two children of any subsequent node by using the following formulae:

2n+1; 2n+2 where n is the index of current

In other words, if we assume the root to be the element at 0th index we can see that its children are present at indices 1 and 2.

Assume that 0 is not a valid element of the tree. Now the class structure would be as follows:

```
Class bst {
        bst(); (5)
        bst(int *, int); //list of element to insert into the tree (5)
        height(); (20)
        median(); //returns the median element in the present in the bst (20)
        insert_new(int); (20)
        delete_ele(int); (10)
        Overloaded “<<” operator so we can print the contents of the tree using cout (20)
}
```

If you need anything extra, add it as a private member of your class

Cout << some_tree << endl; should print the contents of your bst in the following order.

The format for the print is as follows:

Root

Child1 Child2

Children of Child1 | Children of Child2