Discipline of Computing and IT
University of Newcastle

SENG1120/6120 – Semester 1, 2018
Lab 9 (Week 10)

Video guides: https://www.youtube.com/watch?v=5J2khn1hmv4

https://www.youtube.com/watch?v=RdIodAUUVVU

This week's laboratory provides practice in use of recursion for searching. The task specifies the use of an array, but you can use a dynamic structure (with `Node` and `Tree`). Make sure you include full documentation, macro guards, namespaces, etc. with your code.

In lectures we have discussed the structure of `Node`, used in construction of a binary tree. The function of each of these nodes is to store a pointer to the `Item` at that node. An alternate technique for constructing binary trees is to use array cells to provide the function of tree nodes. Thus an array acts as the underlying collection structure for pointers to the items stored in the tree.

The array cell with subscript 1 effectively stores an instance of `Item` at the root of the tree, and the cell with subscript 0 remains unused. The tree is then constructed as follows: the left child of the item stored at cell *n* is stored at the cell with subscript *2n*; the right child of the item stored at cell *n* is stored at the cell with subscript *2n+1*.

In the questions below, v is assumed to be an integer. Array cells that do not store a reference to an item in the tree store a `NULL` reference. You may assume that `Item` is comparable. For this laboratory you may limit the size of the underlying array to 40 cells.

THE UNIVERSITY OF NEWCASTLE AUSTRALIA

1. Write a class template for `ATree` (array-based binary tree) that stores pointers to instances of some type `Item`. The item stored at any node of the tree is bigger than the item stored at its left child, and smaller than the item stored in its right child.

   You will need to implement at least the member functions `add(Item)` (for inserting a provided item into the tree), `search(Item)` which returns the index of the cell that contains `Item`; and `size()` (which reports the number of items stored in the tree) as discussed in lectures.

2. Demonstrate your template class `ATree` by storing, in this order, pointers to the integers 5, 4, 3, 7, 6, 1, 2, 12, 15, 9, 8, 11 into an `ATree` of `int`. Then search for items 6, 15 and 17.

3. Implement `infix()` and use it to print out the contents of your tree.

Good Luck!