1. What are differences between Binary Tree and Binary Search Tree?

Binary tree:

* each node has at most two children

BST:

* In addition to binary tree, the left subtree of a node contains only nodes with values smaller than the root
* The right subtree of a node contains only nodes with values larger than the root.
* The left and right subtree each must also by a binary search tree

2. What happens if you insert an item that is already present in the tree? Should

you care about the number of times an element is present?

* We do not insert an element in the tree there is a node with the same value in the tree.
* We do not care about the number of times an element is present.

3. Is it possible to have a Trinary Search Tree? What would be the implications of

this? Is it worth it?

* Yes, we just add an extra pointer to the node.
* Having a trinary tree means we can store multiple occurrences of the same value in the tree.
* Not really worth it. If we really want to store multiple instances of the same value in the tree, we can simply modify the node structure and use a list to store those value.

4. Using the structure we saw with Heaps, Binary Trees could be implemented

using arrays (or lists). What changes would be made, and what problems could

you foresee?

* It is possible to use an array to implement a BST. We will use the index of the parent to locate the children (i\*2 and i\*2+1 assuming a dummy element is inserted at the beginning).
* The most obvious problem with this approach is, in order to maintain the parent and child relationship in the array, we might need to insert lots of nulls in the array.