Let's say my findBSTNode function looks a bit like:

```
bstNode *findBSTNode(bst *tree, void *value) {
    bstNode *current = tree->root;
    ...other code
}
```

For a bst by itself I would want to do:
```
    tree->comp(current->value, value). Because current->value may be an integer, and value is also an integer.
```

But for a rbt I would want to do:
```
    tree->comp(current->value->value, value). Because current->value is only a wrapper, and current->value->value points to the actual string, integer, or whatever.
```

So how do I accommodate for both?

---

An RBT should have an RBT compare function as well. The compare function should separate the value you want from the RBT value object as a whole and then call the normal compare function on those values. You pass this special comparator to the RBT tree during initialization.