Subject: Exposing access to frequency in bstree
Posted by djkoelz on Mon, 06 Mar 2017 16:28:52 GMT

If the frequency is stored within the "generic value", and not the bstNode, how do we get away with access to the frequency of the value without including the definition of the "generic type" within bstree?

In other words..

The frequency at which a string occurs within the bstree is stored within the string struct itself. Yet the value which we store in the bstree is a void*. Where is the connection made such that the frequency may be accessed?

Subject: Re: Exposing access to frequency in bstree
Posted by SSinischo on Mon, 06 Mar 2017 17:08:20 GMT

When you insert a value that is already in the tree, you main function will pass that to your RBT/VBST class. In those classes, you will check if the value is already in the normal BST. If it is, the RBT/VBST class will grab that value, separate it into its value/color/frequency, increment the frequency, and put it back into the BST.

Subject: Re: Exposing access to frequency in bstree
Posted by djkoelz on Mon, 06 Mar 2017 17:19:56 GMT

So RBT and VBST include the definitions of our generic type? ie. RBT and VBST knows that value has the value, color, and frequency properties?

Subject: Re: Exposing access to frequency in bstree
Posted by davidmccoy on Mon, 06 Mar 2017 17:21:05 GMT

In rbt.c and vbst.c you should define a value struct which contains those details of a rbt value and vbst value. Then when you create a new rbt or vbst, its contained bst will have display and comparator functions which know how to deal with rbt/vbstValues. That bst will have nodes with values that are rbt/vbstValues.