A GREEN Binary Search Tree Class

Rough Draft

The green search tree module

A green search tree (*GST*) is a binary search tree that can handle duplicate values. Here is a conforming *gst.h* file:

```
/*** green binary search tree class ***/

#ifndef __GST_INCLUDED__
#define __GST_INCLUDED__

#include <stdio.h>
#include "tnode.h"

typedef struct gst GST;

extern GST   *newGST(int (*c)(void *,void *));
extern void   setGSTdisplay(GST *t,void (*d)(void *,FILE *));
extern void   setGSTswapper(GST *t,void (*s)(TNODE *,TNODE *));
extern void   setGSTfree(GST *t,void (*)(void *));
extern TNODE *getGSTroot(GST *t);
extern void   setGSTroot(GST *t,TNODE *replacement);
extern void   setGSTsize(GST *t,int s);
extern TNODE *insertGST(GST *t,void *value);
extern void  *findGST(GST *t,void *key);
extern TNODE *locateGST(GST *t,void *key);
extern int    deleteGST(GST *t,void *key);
extern TNODE *swapToLeafGST(GST *t,TNODE *node);
extern void   pruneLeafGST(GST *t,TNODE *leaf);
extern int    sizeGST(GST *t);
extern void   statisticsGST(GST *t,FILE *fp);
extern void   displayGST(GST *t,FILE *fp);
extern int    debugGST(GST *t,int level);
extern void   freeGST(GST *t);

/* extensions of BST */
extern void   *getGSTvalue(TNODE *n);
extern int    freqGST(GST *g,void *key);
extern int    duplicatesGST(GST *g);

#endif
```

Here are some of the behaviors your methods should have. This listing is not exhaustive; you are expected, as a computer scientist, to complete the implementation in the best possible and most logical manner.

- *newGST* - Similar to *newBST*.
- *freqGST* - This method returns the frequency of the searched-for key. If the key is not in the tree, the method should return zero.
- *findGST* - This method returns the value stored with the given key. It returns null if the key is not in the tree.
- *locateGST* - This method returns the tree node holding the searched-for key. If the key is not in the tree, the method should return null.
- *insertGST* - This method attempts to inserts a generic value in the tree. If the value to be inserted is already in the tree, the frequency count of the value in the tree is incremented, the value passed to the insert method is perhaps freed, and a null pointer is returned. If the generic value is not in the tree, it is inserted and the tree node that holds the value is returned. The passed-in generic value should be freed if it is a duplicate *and* a freeing function has been passed to the tree via *setGSTfree*.
- *deleteGST* - The method starts by finding the generic value stored in the tree that matches the given value. If the value is not in the tree, -1 is returned. Otherwise the resulting frequency is returned. If the frequency count of the stored value is greater than one, this method reduces the frequency. If the frequency count is one, however, the *GST* value is removed from the tree (i.e. zero is returned).
- *sizeGST* - This method returns the number of nodes currently in the tree. It should run in amortized constant time.

- *duplicatesGST* - This method returns the number of duplicate values currently in the tree. It should run in amortized constant time. This should be equal to the net number of *GST* insertions minus the number of nodes in the underlying BST.
- *statisticsGST* - This method should display the number of duplicates. Then the method calls the BST statistics method.
- *displayGST* The method calls the tree using a level-order traversal, via the decorated display method of the underlying data structure.
- *debugGST* The method calls the display method of the underlying data structure.
- *getGSTVALUE* This method, when passed a node *n*, extracts the *GST* value. From that extracted value, it returns the generic value that is wrapped by the *GST* value.

Some of these methods will be wrappers for the similarly named *BST* methods, while others will add some functionality. You will need a private function to swap values. It should look something like:

The *swapper* function is passed as the third argument to the *BST* constructor.

The only local includes a *GST* module should have are *gst.h*, *bst.h*, and *tnode.h*.