SortedListPtr SLCreate(CompareFuncT cf)

Creates a new list with head pointing to NULL. Stores the compare function as well. O(1) Runtime

void SLDestroy(SortedListPtr list)

Destroys a list by creating two pointers to traverse the list. At each iteration, the trailing pointer will be freed. Finally the list struct is freed. O(n) Runtime

int SLInsert(SortedListPtr list, void *newObj)

This function assumes that no duplicates are allowed. It will linearly search for the best spot to add in a new node given the data pointer. If equal data is found in the list, the function returns with failure (0). O(n) Worst Case Runtime

int SLRemove(SortedListPtr list, void *newObj)

Linearly searches for the first node that matches data. If a node is found with equal data and marked as removed, function returns with failure to delete as the data isn't available in the list. O(n) Worst Case Runtime

SortedListIteratorPtr SLCreateIterator(SortedListPtr list)

Creates an iterator with the current node marked as null and saves the list. O(1) Runtime

void SLDestroyIterator(SortedListIteratorPtr iterator)

Frees the iterator struct itself and sets to null. O(1) Runtime

void *SLNextItem(SortedListIteratorPtr iterator)

Checks to see if the iterator current value is NULL. If NULL then pass the first element in the lsit. Otherwise move, then return new iterator current value.