Zack Colello and Anna Genke
Programming Assignment 2: Sorted-List

included is sorted-list.h, sorted-list.c, main.c Makefile and testplan.txt

sorted-list.h is a headerfile, with the function delectation of the required functions and the initialization of any structs used in the program. These include the structuers SLNode, SortedList and SortedListIterator.

The Implantation of the functions are in sorted-list.c. The functions are:

SLCreate, which takes in a comparator function, and a destroy function. It creates an instance of a sorted-list and point the compare field to the comparator function and points the destroy field to the destroy function. It does this in O(1) time and only takes the space of the struct.

SLInsert takes in a list and a void* to newObj. It iterates through the linked list and places the node by calling the comparator help by the list. The space it uses is the space of each node. The time used is the time of each call to the compare function which will be called as many times as there are nodes in the list or O(n). to place every node in the list will be O(n^2).

SLDestroy destroys a list by freeing all nodes in the list and then freeing the list object. Destroy takes no space and makes a call to the destroy function n times. It operates in O(n) time.

SLRemove searches through the list using linear search. In uses the compare function stored by the compare function pointer in the list. It calls the compare function n time s and the destroy function once. It doesn't take additional space and operates in O(n) time.

SLCreateIterator creates an instance of a SortedListIterator and points it to the first node in the sorted list. It takes the space of the struct SortedListIterator and works in O(1) time.

SLDestroyIterator frees the space allocated for the iterator and sets it to null. If the Node it is pointing to has a reference counter of zero it also frees the node working in O(1) time.

SLGetItem takes in a SortedListIteratorPtr returns the data held in the node that the SortedListIteratorPtr is pointed to. It uses no extra space and works in O(1) time.

SLNextItem takes in an SortedListIteratorPtr and returns the node's it points to next node, it take up the space on the struct SLNode. It runs in O(1) time and takes O(n) to fill the list.