Starting with method SLCreate, the operation time is O(1) because it is only creating one object to be used as base, the memory usage involves the size SortedList and a Node. The way I came up with this method is based on my struct of SortedList. Next is SLDestroy which has the operation time of O(n)+1 because it must destroy every node in the list then the list itself. The method has no memory usage because it is actually freeing memories. The next method SLInsert has the run time of O(n) as well because it must compare at most all the object in the list in order to determine where to put the object and the memory usage is one extra sizeof Node. SLRemove also has the run time of O(n) because it needs to compare at most all the object in the list to find repetitions which no memory usage because it is taking out one node. The same goes for Iterators the creation method causes only the memory of the Iterator and run time of O(1). The implementation of the Iterator was a bit more difficult due to the fact that one must consider the changes in the sortedList before returning any value.