Abhishek Saha
Nikhil Shenoy
Assignment 2 – CS 214
Prof. Brian Russell

SLCreate has a run time of O(1). It creates a new sorted list in constant time by creating and initializing a new sorted list struct. The memory usage is the size of the SortedListPtr struct.

SLDestroy has a run time of O(n), where n is the number of nodes in the sorted list. It destroys each node in constant time, and destroys n nodes in n units of time. At the end of the function, the memory associated with the sorted list is freed up.

NodeCreate has a run time of O(1). It creates a new node in constant time by creating and initializing a new node The memory usage is the size of the Node struct.

NodeDestroy has a run time of O(1). It destroys one node in constant time by freeing up the memory associated with it.

SLInsert has a run time of O(n), where n is the number of nodes in the sorted list. In the worst case scenario, the function will traverse the entire sorted list and insert a node at the end of the list. There, it will create a node that will require the memory of the size of one Node to be allocated.

SLRemove has a run time of O(n), where n is the number of nodes in the sorted list. In the worst case scenario, the function will traverse the entire sorted list and remove the corresponding node at the end of the list.

SortedCreateIterator has a run time of O(1). It creates a new SortedListIteratorPtr in constant time and uses memory allocated for the size of a SortedListIteratorPtr.

SortedDestroyIterator has a run time of O(1). It destroys a new SortedListIteratorPtr in constant time and frees the memory allocated for the size of a SortedListIteratorPtr.

SLNextItem has a run time of O(1). It moves the iterator one node forward, lowers the reference count by one, and destroys the old node if the refcount is 0. This is done in constant time and does not allocate more memory.