Running Time Analysis:

SLCreate: Allocates and initializes a sorted-list. A sorted-list always starts off with 0 elements when it is created (N = 0). The total size of a sorted-list consists of:
a pointer to the head of the list (8 bytes) + two pointers to the constructor and destructor functions (16 bytes) + size of the sorted-list (N) * size of each node in the sorted-list (16 bytes) = **24 + 16N bytes of memory.**
Complexity of SLCreate: O(1)

SLDestroy: Deletes and Frees a sorted-list. Consists of calling SLRemove() on all of the nodes in the sorted list, in order. Let N be the size of the sorted list:
Complexity = N * best case complexity of SLRemove() (O(1) = O(N)

SLInsert: Inserts the given element in the sorted-list by comparing it to the elements in the rest of the list. Suppose comparing two elements in the sorted list is done in O(1) time. Let N be the number of elements in the list:
Complexity: Worst Case, the element is placed at the end of the list:(N - 1) * O(1) = O(N)

SLRemove: Finds a given element in the sorted-list and removes it from the list. Also frees it if there are no other references to the Node holding that given element. Suppose comparing two elements in the sorted list is done in O(1) time. Let N be the number of elements in the list:
Complexity: Worst Case, the element to be removed is at the end of the list:(N - 1) * O(1) = O(N). Best Case, the element to be removed is the head of the list: O(1)

SLCreateIterator: Given a sorted-list, creates an iterator for that list. Returns a pointer to the created Iterator. Struct consists of two pointers = 16 bytes of memory.
Complexity: O(1)

SLDestroyIterator: Given an iterator, delete it and free its data members if necessary. Because an iterator may link an entire sorted-list, this function can free up to N nodes in the worst case.
Complexity: Worst Case, this frees up N nodes: N * O(1) = O(N)

SLNextItem: Increments the iterator & returns the data member of the node after iterating, if it exists.
Complexity: O(1)

SLGetItem: Returns the data member of the current node, if it exists.
Complexity: O(1)