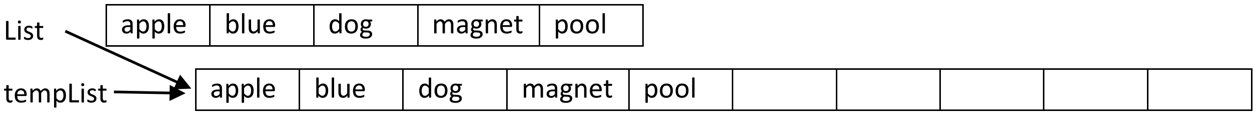
Insert() Method

The insert method inserts the string value that is passed to it in the correct alphabetical order. The first part of the algorithm will loop through the List checking to see if the passed string is already in the list, if it is then a Boolean value is set to true, and if it is not found in the list the value is false. Once the loop is finished the Boolean value is checked and if it is not found in the list then the second part of the method is executed and if the string value is found, then the method exits. If the passed string is not found in the list then the first part of the algorithm is to check to see if the list is full. If the size of the list is equal to the capacity of the list, then a new temporary list is created, which has a capacity of two times of the original list. Then the old list values are copied to the new larger temporary list, and the original list name is pointed to the temporary list, and the list capacity value is changed to twice its value. The final part of the algorithm has four different possibilities. The first is if the passed string value is alphabetically less than the first value in the list, and the values in the list are shifted one position to further into the array, starting from the end of the array to the beginning of the array, and finally putting the passed string value into the first element of the list. The second possibility is if the passed string has to be placed in between two values. A loop is used to go through the list comparing the [i^{th}] and [i^{th} +1] element to the passed string to see if it is alphabetically greater than the [i^{th}] value and alphabetically less than the [i^{th}+1] element. When this spot is found, another loop is completed where the elements from the [i^{th}+1] index to the [size of the list] index are shifted one position further into the list, and the passed string is inserted at the [i^{th}+1] element. The third possibility is the passed string is alphabetically greater than the last string in the list. When this happens, the string is inserted at the end of the list, in the index of the size of the list. The last possible outcome is if the current list is empty. If the size of the list is zero, than the string is inserted at the zero index of the list. Consider the list with a size of five and

List →	apple	blue	dog	magnet	pool

a capacity of 5, and the passed string to the Insert() method is fan. When the method is called, the first thing it will check is if ‘fan’ is already in the list. Since it is not, the Boolean value will be false, and the insert if will be executed. When that if is executed, since the size of the list is equal to the capacity of the list, a new array of string must be created of size twice the original capacity, and the values are copied into the new list, and the old list name points to the new list, the new capacity will have a value of 10.



List	apple	blue	dog	magnet	pool					
tempList →	apple	blue	dog	magnet	pool					

After the new list is made the string is compared to the first and last element in the list to check if it needs to be inserted at the beginning or the end of the list, but since apple < fan < pool, the string has to be inserted in the middle of the array. A loop is used to go through the list to check where the inequality List[i] < fan < List[i+1] is valid. In this case is at index i = 2, List[2] < fan < List[3], dog < fan < magnet. Then the values in the array are changed to insert the string.

First iteration:

List	apple	blue	dog	magnet	pool	pool				

Second iteration:

List	apple	blue	dog	magnet	magnet	pool				

Third iteration:

List	apple	blue	dog	dog	magnet	pool				

After insert() method:

List	apple	blue	fan	dog	magnet	pool				

Once the loop to move the values one position further into the array is finished, which stops are the ith index, which in this case is 2, the passed string is inserted at that index, and the list size value is increased.

Remove() method

The remove() method is used to remove a string from a list of strings. To accomplish this, first the list must be searched to see if the passed string is in the list. If it is not, the method will exit, but if it is then the index of the list where the passed string equals the list value is assigned to a value to be used in the loop to remove the value from the list. To remove the value from the list, a loop is used to move the value in the list from the [ith+1] element to the [ith] element, starting at the index where the passed string is located at. After the loop is completed the last element in the list is assigned an empty value and the size of the list is decreased. Consider the list from the previous example, where we want to remove the string fan.

List	apple	blue	fan	dog	magnet	pool				

The first part of the algorithm is to loop through the array to find the index where the passed string is equal to the list value. In this example the value of List[2] is equal to the passed value fan. After this index is found a loop to move the values one position to the left in the array is executed.

First iteration:

List	apple	blue	dog	dog	magnet	pool				

Second iteration:

List	apple	blue	dog	magnet	magnet	pool				

Third iteration:

List	apple	blue	dog	magnet	pool	pool				

After the loop the last element of the List is assigned an empty value, this is not included in the loop for the case when the list size is equal to the capacity size.

After remove() method:

List	apple	blue	dog	magnet	pool					