Lab 1 – COMP ENG 2SI4

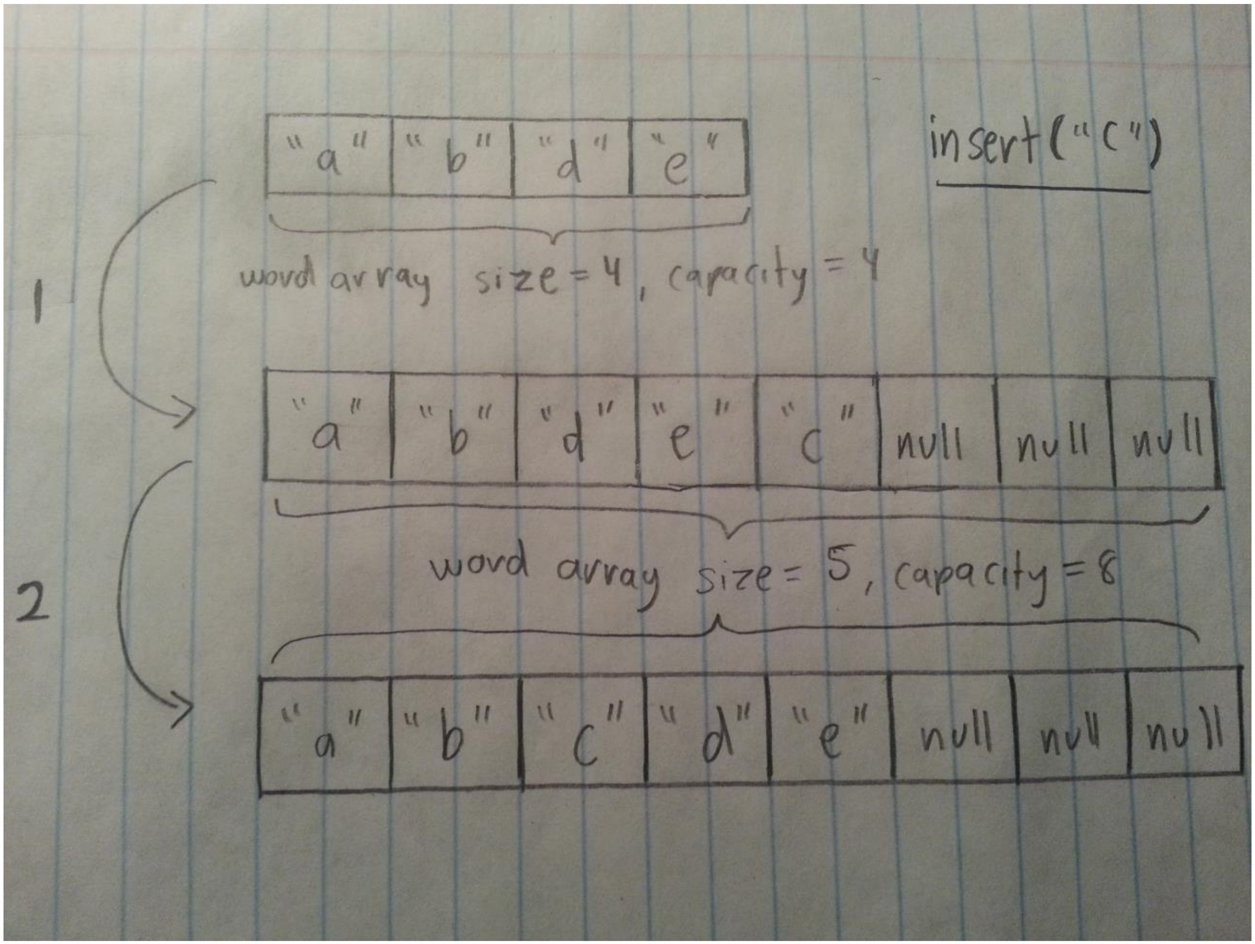
Abilash Logeswaran

Start Date: Jan/19/2015

Halfway Point: Jan/20/2015

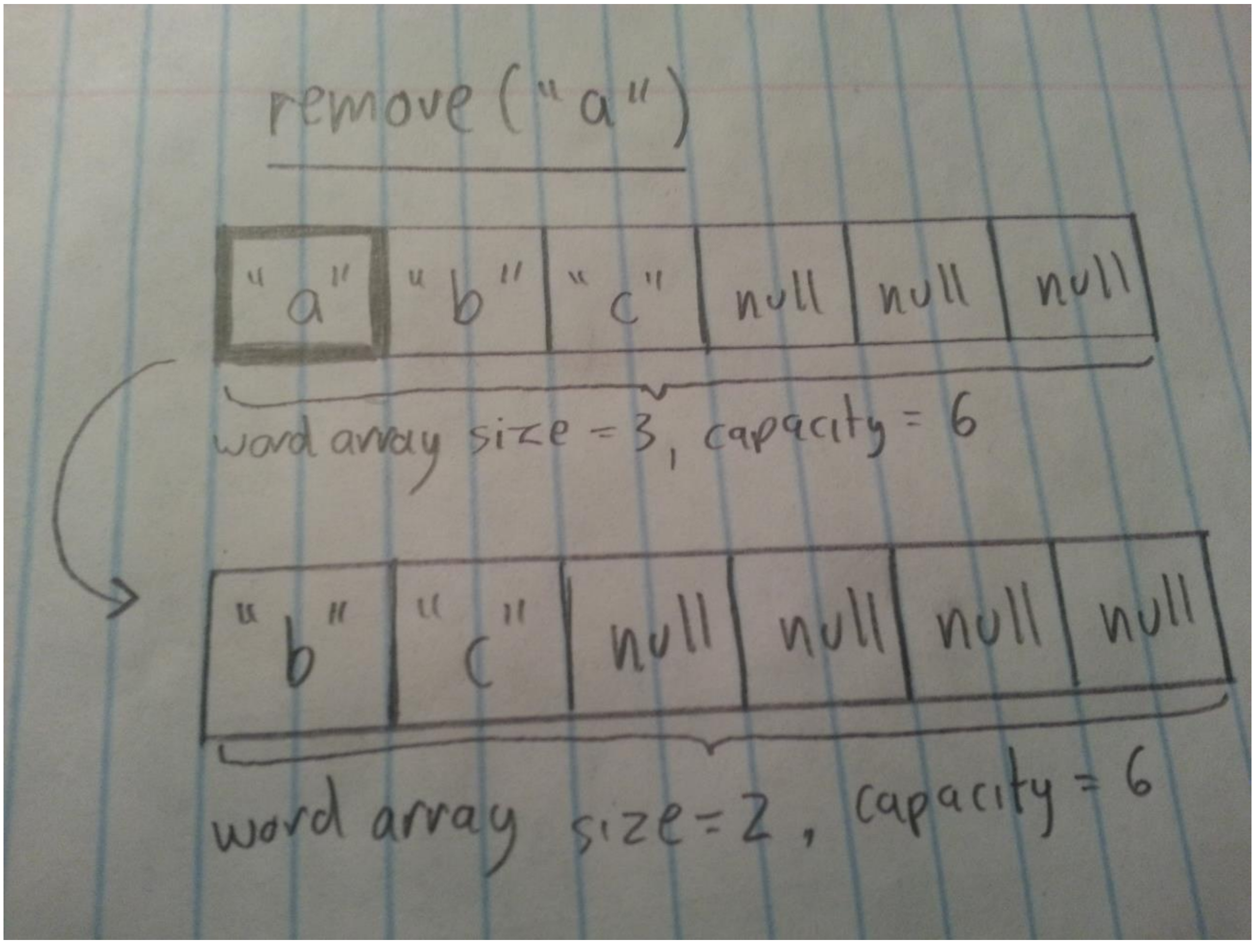
Completion: Jan/22/2015

Insert ():


insert("c")
"a" "b" "d" "e"
word array size=4, capacity=4
1
"a" "b" "d" "e" "c" null null null
word array size=5, capacity=8
2
"a" "b" "c" "d" "e" null null null

The insert function works according to the following algorithm. First the entry being inserted cannot be present in the list already, so using the find() method this is evaluated. The second condition that splits the algorithm into 2 paths is whether the array being inserted into has any other entries; if not, then the word is simply inserted as the first entry. If the size is greater than 0, it is evaluated whether there is space (or capacity) left such that a new word can be inserted. As in the sketch above, if there is insufficient space, the capacity of the list becomes twice the current size and the extra entries are initialized as null. Lastly for the word to be inserted in the correct spot a bubble sort is employed which uses the compareTo() method. First the word is inserted as the last entry (and size is increased by 1), and using a for loop and if condition, this entry is moved further up the list if the compareTo() method indicates a negative value when compared with the adjacent entry, which indicates that the new word is lexographically greater and thus earlier in the alphabet. This continues until a positive value is received at which point the for loop continues executing, but the inserted word does not move further up the list because a positive value indicates it is later in alphabetical order.

Remove ():


remove ("a")
"a"
"b"
"c"
null
null
null
word array size = 3, capacity = 6
"b"
"c"
null
null
null
null
word array size = 2, capacity = 6

The remove() method follows a simpler algorithm as follows. First it confirms if the word to be removed actually exists in the list using the find() function. If this is true, then remove() uses the position of the word as a starting point for a for loop which terminates at size-1 which is the index of the final word in the list. The for loop shifts each entry left, or towards the beginning of the list by making the term in the list equal the term directly to its right. This continues until the very last entry in the list, which at the point has a duplicate, but then its value takes null as that is the value of the right adjacent term. This effectively removes a word from the list.