*CSE 102*

**Collections - I**

**(ArrayList and Vector)**

1. Write a function which takes an ArrayList of Boolean and fills its last half with null references. Note that return-type of your function must be void. You can assume that the size of the parameter list is always even.
2. Write a function which takes an ArrayList of Strings *words* and assigns empty string object to *words*[*i*] where *i* is the smallest integer such that *words*[*i*]is a null reference (of course you cannot access elements of an ArrayList with bracket notation, this is just to demonstrate the point). The list should stay the same if it does not contain any null reference.
3. Write a function which takes an ArrayList of Strings *words* and **adds** just enough strings at the end of it to ensure that the following condition is satisfied: “For every string *w* in *words* the reverse of *w* is also in *words*.” If the condition is already satisfied, no action is needed.

e.g.

complete([“aa”, “aca”, “ba”, “ab”]) would not change anything.

complete([“ab”, “qe”, “eq”]) appends “ba”.

1. Write a function which takes an ArrayList of Strings *words* and **removes** just enough strings from it to ensure that the following condition is satisfied: “For every string *w* in *words* the reverse of *w* is also in *words*.” If the condition is already satisfied, no action is needed.

e.g.

remove([“aa”, “aca”, “ba”, “ab”]) would not change anything.

remove ([“ab”, “qe”, “eq”]) removes “ab”.

1. Write a function which takes a string representing a large corpus of English (a text basically) and returns an ArrayList of Strings consisting of unique English words that appear at least once in this corpus, sorted alphabetically.

getWords(“Once upon a time, there was a queen ruling a cold land. She had a big castle in which there were bright gardens as well as dark dungeons.”) *returns*

[a, as, big, bright, castle, cold, dark, dungeons, gardens, had, in, land, once, queen, ruling, she, there, time, upon, was, well, were, which]