1. The only obstacles I faced in this code writing were considering the cases when n=0 and what the function would return and what the function actually wanted us to return in that case. Also in function rotate left, I realized that we had to check if “pos” was within the array and developed my code accordingly.
2. e[6]={“ben”,”carly”, “ralph”, “nick”, “toby”, “ray”}

h[4] ={“ben”, “carly” , “jeb”, “jeb”, “nick”}

g[2] = {“ralph” , “nick” }

For appendToAll

* (e, -2, lol) – returns -1 because n<0
* (e, 0, lol) – returns 0
* (e, 4, lol) – returns 4 and appends the first 4 elements in the array

For lookup

* (e, -3, carly) – returns -1 because n<0
* (e, 0, carly) – returns -1 because carly not present in the array
* (e, 6, carly) – returns 1 because that’s the position of the word carly

For poistionOfMax

* (e, -3) – returns -1 because n<0
* (e, 0) – returns -1 because empty array
* (e, 6) – returns 4 because toby comes last in alphabetical order

For rotateLeft

* (e, -3, 2) – returns -1 because n<0
* (e, 6, 7) – returns -1 because pos>n
* (e, 6, -2) – returns -1 because pos<0
* (e, 6, 2) – returns 2 and rotates left from position 2

For countRuns

* (h, -3) – returns -1 because n<0
* (h, 5) – returns 4
* (h, 0) – returns 0 because no elements

For flip

* (e, -3) – returns -1 because n<0
* (e, 0) – returns 0 because n=0
* (e, 6) – returns 6 and reverses the array

For differ

* (e, -3, h, 4) – returns -1 because n1<0
* (e, 6, h, -4) – returns -1 because n2<0
* (e, 0, h, 4) – returns 0 because they differ from position 0
* (e, 6, h, 0) – returns 0 because they differ from position 0
* (e, 6, h, 4) – returns 2

For subsequence

* (e, -3, g, 2) – returns -1 because n1<0
* (e, 6, g, -3) – returns -1 because n2<0
* (e, 0, g, 2) – returns -1 because subsequence not found in e
* (e, 6, g, 0) – returns 0 because no element is a subsequence of first e from position 0
* (e, 6, g, 2) – returns 2 because subsequence found at position 2

For lookupAny

* (e, -3, g, 2) – returns -1 because n1<0
* (e, 6, g, -3) – returns -1 because n2<0
* (e, 0, g, 2) – returns -1 because no match found in e
* (e, 6, g, 0) – returns -1 because e has nothing to look for in g and thus no match
* (e, 6, g, 2) – returns 2

For split

* (h, -1, car) – returns -1 because n<0
* (h, 0, car) – returns 0 because n=0 , no splitting occurs
* (h, 4, car) – returns 1 because carly > car
* (h, 4, zed) – returns 4 because nothing >splitter