a. One obstacle I overcame is how to write cout to test my functions whether it works correctly for the program. Apparently, I struggled understanding how to make comparison between two array. At first , I was thinking to take first letter and then compare with every other letter in the string, which I wrote a lot of the tedious codes for that cases. This took too long to finish the first function. After reading the textbook, I was able to understand that c++ has already taken into account. All the letters are following increasing order and rate from small to large as the letter is increasing. The process of writing out the function has become much easier. I was able to write out the correct function and test them correctly with cout.

b.

const string array[4]={"a", "z", "z","a"};

const string array[4]={"a", "l", "c","u"};

cout << locateMinimum(array,4) << endl;

this case will test the first function whether there are repeated string in the array.

const string arrayi[4]={"apple", "apple", "apple","udd"};

cout << findLastOccurrence(arrayi,4,"apple" ) << endl;

this cout will test whether the function will give us correct value of string that is given.

string arrayh[8]={"b", "o", "a","u","p","a","g","l"};

cout << flipAround(arrayh,8) << endl;

this case will test whether this function works correctly to reverse the order and still come out with the correct index value.

Repeated upper and lower cases;

const string arrayk[8]={"a","b","c","d","e","j","g","g"};

cout << hasNoDuplicates(arrayk, 8) << endl;

this case will test whether the function that has duplicate value or no, if there is duplicate value return false otherwise return true.

const string arrayk[8]={"a","a","c","d","e","j","g","g"};

const string arrayk[8]={"a","b","c","d","e","j","g","g"};

cout << isincreasingorder(arrayk, 8) << endl;

this will check whether the string is in increasing order. This can be repeated letter in the beginning and at the end .

const string array1[5]={"e", "r", "a","s", "t"};

const string array2[3]= {"b", "r", "a"};

string resultString[6];

int resultSize;

unionWithNoDuplicates(array1, 5, array2, 3, resultString, resultSize);

for (int i = 0; i < resultSize; i++) {

cout << resultString[i] << endl;