1. A brief description of notable obstacles you overcame.

Because the project requires us to write functions that is somehow independent to each other, it is relatively easy for me to develop. The most often obstacles I encountered is the problem of out of boundary, which I may accidentally access the index that is larger than the size of the array. Another problem I encountered is how to deal with the special inputs and give them proper return value, like the case with n is 0.

1. A list of the test data

string h[7] = { "martha", "mark", "joe", "susan", "", "kamala", "lindsey" };

Case of notwithstanding

appendToAll(h,-2,"!")

Normal case

appendToAll(h,2,"!")

Case with n=0

appendToAll(h,0,"!")

h[7] = { "martha", "mark", "joe", "susan", "", "kamala", "lindsey" };

Case of notwithstanding

lookup(h,-2,"mark")

Normal case

lookup(h,7,"mark")

Normal case

lookup(h,7,"marky")

Case with n=0

lookup(h,0,"marky")

h[7] = { "martha", "mark", "joe", "susan", "", "kamala", "lindsey" };

Case of notwithstanding

positionOfMax(h,-1)

positionOfMax(h,0)

Normal case

positionOfMax(h,7)

positionOfMax(h,3)

Case with same max element

h[7] = { "martha", "mark", "joe", "susan", " susan ", "kamala", "lindsey" };

positionOfMax(h,7)

h[7] = { "martha", "mark", "joe", "susan", "", "kamala", "lindsey" };

Case of notwithstanding

rotateLeft(h,7,-1)

rotateLeft(h,7,8)

rotateLeft(h,0,-1)

rotateLeft(h,0,0)

Normal case

rotateLeft(h,7,5)

rotateLeft(h,7,6)

h[7] = { "martha", "mark", " mark ", " mark", "kamala", "kamala", "lindsey" };

Case of notwithstanding

countRuns(h,-2)

Normal case

countRuns(h,0)

countRuns(h,4)

countRuns(h,7)

string h[7] = { "martha", "mark", "joe", "susan", "", "kamala", "lindsey" };

Case of notwithstanding

flip(h,-2)

Normal case

flip(h,2)

flip(h,7)

string c[7] = { "martha", "mark", "joe", "susan", "", "kamala", "lindsey" };

string d[7] = { "martha", "mark", "joe", "lindsey", "kamala", "joe","lindsey" };

Case of notwithstanding

differ(c,-1,d,0)

differ(c,-1,d,-1)

Normal case

differ(c, 3,d,3)

differ(c,7,d,7)

differ(c,3,d,7)

string h[7] = { "martha", "mark", "joe", "susan", "", "kamala", "lindsey" };

string i[3] = { "martha", "mark", "joe"};

string j[4] = { "martha", "mark", "joe", "lindsey" };

Case of notwithstanding

subsequence(h,-1,i,0)

subsequence(h,-1,i,-1)

Normal case

subsequence(h,7,i,0)

subsequence(h,0,i,0)

subsequence(h,7,i,3)

subsequence(h,7,j,4)

subsequence(h,7,j,3)

string h[7] = { "martha", "mark", "", "susan", "joe", "kamala", "lindsey" };

string i[3] = { "mike", "jake", "joe"};

Case of notwithstanding

lookUpAny(h,-1,i,-1)

lookUpAny(h,0,i,0)

Normal case

lookUpAny(h,7,i,3)

lookUpAny(h,3,i,3)

lookUpAny(h,7,i,2)

string h[7] = { "martha", "mark", "kim", "susan", "joe", "kamala", "lindsey" };

Case of notwithstanding

divide(h,-1, "mark")

Normal case

divide(h,7, "mark")

divide(h,6, "mark")

divide(h,7, "max")

divide(h,7, "apple")