Melody Chen

CS 31

#705120273

Project 4 Report

One of the obstacles I faced doing Project 4 was when I did not read the specs carefully and did not return the right values for invalid sizes entered. Occasionally I returned -1 or 0, which messed up my program when I was trying to pass them through test cases. I also realized after finishing the project that for certain functions 0 should be returned when the function cannot fulfill its purpose, such as the function replaceAllCharacters.

string test1[8] = {

"ellie", "josh", "janis", "josh",

"angela", "joanna", "angela", "ryan" };

string test2[8] = {

"","", ""};

*//for has duplicates*

assert(hasDuplicates(test1, 3)==**false**); *//tests if size limits range*

assert(hasDuplicates(test1, 4)==**true**); *//tests for duplicates*

assert(hasDuplicates(test1, 8)==**true**); *//tests for multiple duplicates*

assert(hasDuplicates(test1, -5)==**false**); *//tests for negative size*

assert(hasDuplicates(test1, 0)==**false**); *//tests for zero size*

assert(hasDuplicates(test2, 2)==**true**); *//tests for empty strings*

assert(hasDuplicates(test2, 1)==**false**); *//tests for only one string in array*

string test3[8] = {"el44444","janis222", "josh345"};

*//for count all digits*

assert(countAllDigits(test3, 3)==11); *//tests if counts right number*

assert(countAllDigits(test3, 2)==8); *//tests if counts right number*

assert(countAllDigits(test3, 1)==5); *//tests if counts right number*

assert(countAllDigits(test3, 0)==-1); *//when size is 0*

assert(countAllDigits(test3, -5)==-1); *//when size is negative*

assert(countAllDigits(test1, 8)==0); *//when there are no numbers*

assert(countAllDigits(test2, 3)==0); *//when there are no numbers*

string test4[6] = {"zshushus","z","a","ZZZ", "BBa", "A"};

string test5[5] = {"apple","apple","a","ZZZ"};

*//for IsInDecreasingOrder*

assert(isInDecreasingOrder(test4, 6)==**true**); *//tests if works correctly*

assert(isInDecreasingOrder(test4, 1)==**true**); *//test for array size of 1*

assert(isInDecreasingOrder(test4, 0)==**true**); *//tests for array size of 0*

assert(isInDecreasingOrder(test4, -5)==**false**); *//tests for negative numbers*

assert(isInDecreasingOrder(test4, -1)==**false**); *//tests for negative numbers*

assert(isInDecreasingOrder(test3, 3)==**false**); *//tests for not decreasing order*

assert(isInDecreasingOrder(test5, 4)==**false**); *//tests for same strings*

*//for shiftRight*

assert(shiftRight(test5, -5, 2, "trash")==-1);*//test for false size*

assert(shiftRight(test5, 4, 0, "trash")==0);*//tests for amount is 0*

assert(shiftRight(test5, 0, 0, "trash")==0);*//tests for amount and size is 0*

assert(shiftRight(test5, 0, -5, "trash")==-1);*//tests for amount is negative*

assert(shiftRight(test5, 4, 1, "trash")==1);*//tests for returning right value*

assert(test5[0]=="trash");*//tests if replace with correct string*

assert(test5[2]=="apple");*//tests if actually shifted*

assert(test5[3]=="a");*//tests if shifted*

assert(shiftRight(test4, 4, 4, "trash")==4);*//tests for returning right value*

assert(test4[3]=="trash");*//tests if shifted*

assert(test4[4]=="BBa");*//tests if string not included will not be shifted*

assert(shiftRight(test3, 3, 0, "trash")==0);*//tests for returning right value*

assert(test3[0]=="el44444");*//tests if string not included will not be shifted*

string test6[4]={"hello", "???", "hey", "hello"};

string test7[4]={"", "", "hey", "hello"};

*//for find*

assert(find(test6,3,"???")==1); *//tests if find at correct index*

assert(find(test6,3,"??")==-1); *//tests if cannot find string*

assert(find(test6,0,"??")==-1); *//tests if size is 0*

assert(find(test6,-5,"??")==-1); *//tests if size is negative*

assert(find(test6,4,"hello")==0); *//tests if find first string*

assert(find(test6,4,"hey")==2); *//tests if find first string*

*//forReplaceAllCharacters*

assert(replaceAllCharacters(test6, 4, 'h', '!')==3); *//returns correct number of chars replaced*

assert(test6[0][0]=='!'); *//tests if char is replaced correctly*

assert(test6[2][0]=='!'); *//tests if char is replaced correctly*

assert(test6[3][0]=='!'); *//tests if char is replaced correctly*

assert(replaceAllCharacters(test6, -5, 'h', '!')==-1); *//if size is negative*

assert(replaceAllCharacters(test6, 0, 'h', '!')==-1); *//if size is zero*

assert(replaceAllCharacters(test6, 4, 'a', '!')==0); *//if char findLetter does not exist*

assert(replaceAllCharacters(test7, 4, 'a', '!')==0); *//if char findLetter does not exist*

assert(test7[2]=="hey"); *//if strings unchanged remain the same*

cerr<<"Passed all test cases."<<endl;

**return** 0;