Report

Jingjing Nie (UID:304567417)

a. Some of the notable obstacles I overcame were to use loops to check each element in the array and to ensure that the use of multiple or composite loops is logical and functions well. Also, I found merging two arrays into one while removing the duplicated element pretty challenging as well.

b. the test cases I can use include:

string a[5] = {“angela”, “bella”, “chris”, “Dalia”, “emlia”}

string b[4] = {“delta”, “alaska”, “virginia”, “delta”}

string output [8] = {“ ”, “ ”, “ ”, “ ”, “ ”, “ ”, “ ”, “ ”,};

int outputSize = 8;

locateMinimum (a, 3) should return 3, since capital D should be sorted as the minimum element;

locateMinimum (b, 0) should return -1, since int n equal to 0.

findLastOccurence (a, 5, “bella”) should return 1, since “bella” has index 1;

findLastOccurence (b, 4, “doll”) should return -1, since “doll” is not found in b;

hasNoDuplicates (a, 5) should return true, since all are different in array a;

hasNoDuplicates (b, 4) should return true, since “delta” is duplicated in b;

isInIncreasingOrder (a, 3) should return true, since the first three strings are in increasing order;

isInIncreasingOrder (a, 5) should return flase, since the strings are not in increasing order;

unionWithNoDuplicates (a, 5, b, 4, output, outputSize) should produce outputSize as 8, since there are 8 different strings in two arrays combined;

unionWithNoDuplicates (a, 2, b, 4, output, outputSize) should produce outputSize as 5, since there are 5 different strings in two arrays combined;

shiftRight (a, 5, 3, “well”) should return 2, since there are 2 original strings left;

shiftRight (b, 4, 4, “well”) should return 0, since no original strings are left;

flipAround (a, 5) should return 2, since the process should be carried out twice;

flipAround (b, 0) should return 0, since the process should be carried out 0 time.