Computer Science 31 Kitu Komya

Project 4 UID: 404-491-375

**Notable Obstacles**

Before I began writing the functions, I had a skeleton outline of them but didn’t have them return anything. As soon as I compiled, I got so many errors because you need to by default at least have it return something! Also, my int main function I realized does not work unless it is after all of the functions (aka once the functions are declared only then may they be called). Most of the functions were very self-explanatory actually, but for the flipAround and hasNoDuplicates function, because there were nested for loops, it took me some time to write out what was actually happening on paper before I could implement them. Sometimes, for some of the functions I would place the default return option (either true or false) in the wrong place in the function, and by default I would always get that. It took some debugging and some careful, methodical analyzation to realize where the return statements needed to go.

**Test Cases**

where string people[5] = { "samwell", "jon", "margaery", "daemerys", "tyrion" };

where string people2[5] = { "samwell", "jon", "daemerys", "daemerys", "tyrion" };

where string people3[5] = { "", "", "", "", "" };

where string people4[5] = { "amwell", "jon", "daemerys", "daemerys", "tyrion" };

locateMinimum

assert(locateMinimum(people, 5) == 3); // in array

assert(locateMinimum(people2, 5) == 2); // one duplicate

assert(locateMinimum(people3, 5) == 0); // empty string

assert(locateMinimum(people, -3) == -1); // negative n value

findLastOccurrence

assert(findLastOccurrence(people, 5, “samwell”) == 0); // in array

assert(findLastOccurrence(people, 5, “testing”) == -1); // not in array

assert(findLastOccurrence(people2, 5, “daemerys”) == 2); // one duplicate

assert(findLastOccurrence(people, -10, “daemerys”) == -1); // negative n value

flipAround

assert(flipArround(people, 5) == 2); // in array

assert(flipArround(people, -10) == -1); // negative n value

assert(flipArround(people3, 5) == 0); // empty array

hasNoDuplicates

assert(hasNoDuplicates(people, 5) == true); // no duplicates

assert(hasNoDuplicates(people2, 5) == false); // one duplicate

assert(hasNoDuplicates(people, -7) == false); // negative n value

unionWithNoDuplicates

assert(unionWithNoDuplicates(people, 5, people2, 5, result, size)); // in array. won’t return anything

shiftRight

assert(shiftRight(people, 5, 3, “foo”) == 2); // in array

assert(shiftRight(people, 3, 5, “foo”) == -1); // amount is greater than n

assert(shiftRight(people, -5, 3, “foo”) == -1); // negative n value

isInIncreasingOrder

assert(isInIncreasingOrder(people, 5) == false) // not in order

assert(isInIncreasingOrder(people4, 3) == true) // in order

assert(isInIncreasingOrder(people4, -34) == false) // negative n value