

*Scott Park : 705844713*

**A.** The first two functions which I started working on were the **locateMaximum** and **hasNoCapitals** functions, which were fairly simple and clear for me to code. With the **locateMaximum** function, it was clear that a for loop could solve the problem, and with the help of my experience in CS30, **hasNoCapitals** was also easy to recognize as a for loop nested in another for loop. The only problems I faced were minor syntax errors occurring when testing these two functions.

However, the third function which I decided to tackle was the **shiftLeft** function. The actual output value of the code was extremely simple, if *amount* > *n*, return *n*, and otherwise return *amount.* The hard part was actually shifting the code left and using placeholders. Mainly, keeping track of the index value which I was trying to use, as well as the other number values involved with n and the amount was very tricky, and required some trial and error to finally reach my solution with a for loop, and a simple if-else statement.

The hardest function for me to code was the **countFloatingPointValues** function. The process of working through the code was simple to begin with. I found similarities with other functions I already coded, such as the if-else statement with when n<= 0, as well as the use of another nested for-loop. After this however, there were many obstacles which I had to work through because my code was not running correctly, producing countless logical errors in my code. For one, I had to figure out where to declare the variables I was using, whether it be at the top of the function, or within a for loop so the scope could be smaller. After I figured that out, I then faced another problem when mistakenly using || where I should have used &&. When my code wasn’t running correctly, I began to scramble and try to change everything in my code, but after I calmed down and walked through the code slowly, I realized the problem.

**B.** **TEST DATA**

string a[7] = {"ILove", "You", "Professor Huang","And", "IReally", "Like", "Zombies"};

string b[5] = { "1000", "20000","300","400","5000" };

string c[7] = { "Chloe","Kim","is","a great athlete","120","zap","900" };

string d[6] = { "20.1", ".12","2.2.2",".", "+12","00000.1"};

string e[3] = { "alphabet","soup","YUMMY" };

assert(locateMaximum(a,4) == 1);

* *Standard locateMaximum test case to see if it runs correctly*

assert(locateMaximum(a, -2) == -1);

* *Test when n is less than or equal to 0*

assert(locateMaximum(a, 7) == 6);

* *Another Standard test case, with change of n from previous*

assert(locateMaximum(b, 3) == 2);

* *Instead of letters, compare digit strings using locateMaximum*

assert(locateMaximum(c, 7) == 5);

* *Both digit and letter strings in the array*

assert(countFloatingPointValues(a, 7) == 0);

* *Test when there aren't Floating Point Values*

assert(countFloatingPointValues(b, -1) == -1);

* *Test when n is less than or equal to 0*

assert(countFloatingPointValues(c, 7) == 2);

* *Standard case with only 2 floating points*

assert(countFloatingPointValues(d, 6) == 4);

* *Multiple decimal points in a string and use of characters such as +*

assert(countFloatingPointValues(d, 3) == 2);

* *Change n from previous test case*

assert(hasNoCapitals(a, 7) == false);

* *Test Case with capital letters present*

assert(hasNoCapitals(b, 4) == true);

* *Test case with only digits*

assert(hasNoCapitals(c, 0) == true);

* *When n is less than or equal to 0*

assert(hasNoCapitals(e, 3) == false);

* *Only one string/element with capitals present*

assert(hasNoCapitals(e, 2) == true);

* *Change n from previous test so capitals aren’t present*

assert(shiftLeft(a, 5, 2, "WAGABOO") == 2);

* *Standard test where n is greater than amount value*

assert(shiftLeft(b, 3, 4, "Fill") == 3);

* *Test when amount value is greater than the n value*

assert(shiftLeft(c, 2, 20, "yuh") == 2);

* *Test when the amount value exceeds the number of elements present in the string*

assert(shiftLeft(d, -1, 10, "poop") == -1);

* *When n is less than or equal to 0*

assert(shiftLeft(e, 2, -3, "ok") == -1);

* *When amount is less than or equal to 0*

*\*In order to test if the actual string was changed by the shiftLeft function, I also slowly walked through each test case by hand, keeping track of if the string produced from my code would equal the expected result\**