Project 4 FAQ

1. **How do I begin? Do I have to implement every function you asked before I run my code?**

   Throw away a mindset that says you have to figure out how all the functions will work before you type in any code. Start with one function and get that working — that will clear up a lot of misconceptions you may have.

   If you're *really* stuck, start with this program that solves a problem that is different, but related, and mutate it into what you want:

```
        #include <iostream>
        #include <string>
        #include <cassert>
        using namespace std;

        // Return the position of the first element that is not <= the one
  // that follows it.  Return -1 if there are no such elements.
        int findFirstDisorder(const string a[], int n)
        {
            for (int k = 0; k < n-1; k++)
          if (a[k] > a[k+1])
              return k;
            return -1;
        }

        int main()
        {
            string h[5] = { "daenerys", "jon", "tyrion", "samwell",
"margaery" };
            assert(findFirstDisorder(h, 5) == 2);
            assert(findFirstDisorder(h, 4) == 2);
            assert(findFirstDisorder(h, 3) == -1);
            assert(findFirstDisorder(h, 2) == -1);
            assert(findFirstDisorder(h, 1) == -1);
            assert(findFirstDisorder(h, 0) == -1); // No disorder in empty
array
            // There should be another test that passes a bad argument
            cout << "All tests succeeded" << endl;
        }
```

2. **I get apparently random results when I try to run some little tests of the comparison operators on strings. For example, a test like `"abcde" < "abcxyz"` sometimes returns false. What's going on?**

To get meaningful results, make sure that at least one of the two objects you are comparing are declared to be of the string type, like `s`,`t`, `a[0]`, and `a[1]` below:

```
string s = "hello";
string t = "help";
string a[2] = { "helping", "hello" };
```

The following are all true:

```
s < t           // The h's, e's, and first l's match, but s has
                // an 'l' where t has a 'p', and 'l' < 'p'.
s < "help"      // True for the same reason.
t < a[0]        // The first four characters match, then t runs
                // out while a[0] has more.
"help" < a[0]   // True for the same reason.
s == a[1]       // All 5 characters match, and both strings end at
                // the same time.
s == "hello"    // True for the same reason
```

What you cannot do predictably is compare two character string literals: `"hello" < "help"` may be true or false. A test like this unfortunately will compile. For technical reasons, what are being compared are not the characters in the literals, but the addresses at which the literals are stored, so the result depends on where the compiler chooses to place the two.

3. **Blah blah blah should I set it to the empty string?**

   There is absolutely nothing in the spec that specially distinguishes the empty string (i.e., the string `""`). For example, `locateMaximum` doesn't treat the empty string as an error, etc. The empty string is just another string, no more special than `"tyrion"` or `"daenerys"`.

4. **What should my functions do if n equals 0?**

   Unless otherwise noted, passing 0 to the function as the array size is not itself an error; it merely indicates the function should examine no elements of the array. Similarly, if you are passed a value less than 0, you should not examine any element of the array at all. I tried my best to be consistent in describing the different problems you are supposed to work on.