Writing Code with Arrays

When we write code with arrays, we need to look for patterns in where we access the array. A common pattern is to access each element of the array in order (the *0th*, then the *1st*, *etc.*). Such a pattern generally lends itself naturally to counting over the elements of the array with a for loop. However, we might have other patterns—as always, we should work the problem ourselves, write down what we did, and look for the patterns.

If we have complicated data *(e.g.*, arrays of structs which have arrays inside them) it is very important to think carefully about how we can name the particular box we want to manipulate. For problems with complex structured data, drawing a diagram with the appropriate pointers and arrays can be an important aspect of working an example of the problem ourselves (Step 1). Then, when we write down exactly what we did (Step 2), we can think carefully about how we can name each box that we manipulate. Remember that sometimes a box will have multiple names, in which case we need to think about *how* we picked that box—what "route" we took to get to it, to figure out what name is most appropriate. If we are unsure, we should make a note of the other names for the box that we can think of; then in Step 3, we may realize that we prefer another way to name the same box, as it creates a more consistent pattern with other "almost repetitive" parts of our algorithm.

Index of Largest Element

In our motivating example (breaking simple cryptographic systems), we discussed solving the problem of finding the largest of 26 numbers. Now that we know about arrays, we can implement a much better version of this function. We can also make our function slightly more general—instead of only operating on 26 pieces of data, we can make it work on an array of any size. The video "Index of Largest Element" walks through this example.

Closest Point

As another example, we can return to the Closest Point algorithm we considered in a previous course. When we first considered this example, we worked through the design of the algorithm for this problem. However, at that time, we did not know how to write any C code, so we stopped there. Now, however, we are ready to complete that example. The video "Closest Point Step-Through" walks through the translation of this algorithm to code.