Notes on Module 6

Naomi Nishimura

As mentioned in class, I'm not happy with the way the marble multiples example appears in the notes or with how I explained it in class. Here's an explanation of what I think should be there instead.

To choose between counting up and counting down, imagine all the numbers lined up and our choosing to look for a good candidate either going from smallest to largest or largest to smallest. If we're trying to find the smallest good candidate, it makes sense to start with the smallest number and keep looking at larger and larger numbers until we find a good one. Conversely, if we're trying to find the largest good candidate, it makes sense to start with the largest number and keep looking at smaller and smaller until we find a good one.

Since we want the smallest good number, we consider items in the range from (*max n1 n2 n3*) to (∗ *n1 n2 n3*) by counting up. This means that we want to use the counting up template, where we count up from *n* to a base case *b*, where here the base case should be (∗ *n1 n2 n3*). We can't just set *b* to this value, since our function *is-good?* uses *n1*, *n2*, and *n3* all as arguments.

We choose to modify the counting up function to have four parameters, with *n1*, *n2*, and *n3* taking the place of *b*. The modified *my-upto-fun* template gives us:

(**define** (*range-marbles n n1 n2 n3*)
  (**cond**
    [(= *n* (∗ *n1 n2 n3*)) ... (∗ *n1 n2 n3*) ...]
    [**else** ... (*range-marbles* (*add1 n*) *n1 n2 n3*) ...]

Our function finds the smallest "good" number in the range from *n* to (∗ *n1 n2 n3*).

For the base case, the range is from (∗ *n1 n2 n3*) to (∗ *n1 n2 n3*). Since (∗ *n1 n2 n3*) is trivially a "good" number, we produce (∗ *n1 n2 n3*).

If instead *n* is less than (∗ *n1 n2 n3*), we have two cases depending on whether or not *n* is good. If it is good, we produce *n*. Otherwise, we produce what we get from the template.

This gives us the following function definition:

(**define** (*range-marbles n n1 n2 n3*)
  (**cond**
    [(= *n* (∗ *n1 n2 n3*)) (∗ *n1 n2 n3*)]
    [(*is-good? n n1 n2 n3*) *n*]
    [**else** (*range-marbles* (*add1 n*) *n1 n2 n3*)]))

The function given in the notes is also correct, since when *n* equals (∗ *n1 n2 n3*), (*is-good? n n1 n2 n3*) will produce *true*. The function given in the notes is shorter, but not necessarily better, as it hides the base case.

There are no strict rules about what is a better way of writing a function; always think about what will be most expressive and clear. When writing assignments for the course, err on the side of making it clear that you understand what template you are using.

Finally, we observe that since we have more parameters than specified, we use a wrapper function to change the number of parameters and start the function at the minimum value of (*max n1 n2 n3*).