Common Confusion 12
Hardcoding
Matthew Woodring

The line between hardcoding and not hardcoding is a thin one. Whether your code is considered hardcoded or not is entirely dependent on the problem statement your code solves.

Hardcoding is defined as writing code that will only work for specific inputs when the code either explicitly or implicitly says it must work for all inputs. To better understand this, consider the following problem statement:

“For a [5x5] matrix, return the value in the center.”

Since the code explicitly says a “[5x5] matrix” the following code for a [5x5] matrix ‘M1’ would *not* be considered hardcoding:

“answer = M1(3, 3)”

This is because center value of any [5x5] matrix is guaranteed to be at the index (3, 3). Since the problem explicitly stated the input will always be a [5x5] matrix, using numerical indexing is not hardcoding. Now, consider the original problem statement said:

“For any matrix of size [nxn], where ‘n’ is odd, return the value in the center.”

Now, if we did the following code for a matrix ‘M2’, it *would* be considered hardcoding:

“answer = M2(3, 3)”

This is because the center of the matrix is *not* guaranteed to be at the index (3, 3). If ‘n’ equaled ‘7’, which would result in a [7x7] matrix, the center would be located at the index (4, 4). Clearly, this is hardcoding since the problem explicitly stated that the function must work for all inputs.

Deciding whether something is hardcoding or not is a skill that takes time to develop. In general, try to avoid anything that could be considered hardcoding *unless* you are certain it is not hardcoding. The use of hardcoding is one of the easiest ways to lose points on MA’s and exams. Once you do enough practice problems, it will become easy to detect when hardcoding is occurring.