# Project 1 Final Report

This project, upon initial cursory glance, appeared to be nothing more than a simple programming assignment that could be bashed out in a few hours after work one evening. After all, we’ve been programming doubly linked lists for at least a couple years now and it should be nothing new. My 6 a.m. epiphany after staring at code for hours proved that this was no easy task and should have probably been taken more seriously from the get go.

I consider the add\_list function and the find\_item function to be the two functions that are particularly “critical”, judging (if anything) by the lines of code that went in to each of them. These are the functions that have to traverse the entire list to really get anything done, especially the add\_list function that literally has to “touch” nearly every item. Considering there are only 6 functions in this program, it’s hard to pick anything as “more critical” than the others; every function in this program had a specific thing that it was trying to accomplish, and they all do so with a sense of purpose.

I should note that I left out chunks of code that were included in project1.c, but for good reason. Given the scenario (and this is noted in the comments, by the way), we are never given a key to find that does not exist, so there is no reason to implement that. If this were an actual program that would need to be used, then yes, that would be needed. Here, however, it would just serve to muddle things up and, in fact, would never be reached in execution whatsoever, so I omitted that section. I believe I did something like that elsewhere, as well, for the exact same reasons. Given the current assignment, there is no need for code that will never be reached in execution.

Also, I have found that perhaps debug strings are not *always* the absolute best thing in the world when it comes to debugging. For a C/C++ program, sure, debug strings are great and can tell you exactly when things take a turn for the worse, but in MIPS it’s a different story. I had instances where I’d get errors for trying to include debug strings (for whatever reason, I could never figure out exactly why), and had to *slow down the execution time of the program* to actually *watch* what it was doing. Never before have I been able to actually *see* what a program is doing in real-time. This proved to be the greatest method of overcoming any problems in my code.

Real-time code analysis (fancy buzzwords) notwithstanding, the other method of overcoming problems involved actually reading and understanding what was going on in project1.c. I recall a moment where my add\_item function wasn’t working, regardless of what I would do. It wasn’t until I went through the MIPS and C code *line-by-line* was I able to find the fatal flaw (a forgotten *jump*, by the way).

As higher-level language programmers (by default, it seems), there are a lot of things that we take for granted in our everyday programming exercises and whatnot. MIPS seems like everything is laid bare and we have to *actually* *know what we’re doing* to be able to accomplish anything at all. A program like this would have been much easier in C/C++, and it makes me appreciate the fact that we have those languages to work with. This was a tough assignment, that’s for sure, but I feel like I’ve actually accomplished something in completing it.