CST250 Fall 2015 – Final Project Reflection – John Abbott

What techniques and strategies did you use to write and optimizations you code? How did your teammates help out? Did you have any problems working as a team? Focus on what you got out of the experience (250-300 words spread across 2-3 paragraphs).

Rather than write the whole thing as a group, we decided to divide the work by assigning one of the three sorting methods to each team member. I got the bubble sort. I started out by typing out the code from the example in the text book to see if it would work as is. I expected it would not as there seem to be some differences between PLP and MIPS assembly. I didn't really know what to do next so I left that code alone for the time being and started looking around on the internet for more information about the bubble sort algorithm and examples of code in different languages. It turns out that this is a pretty common method and is often used as an example for teaching students about sorting algorithms so there were a lot of examples in pseudocode, C++, Java, etc.

Armed with all of this, I wrote out in English what I thought each part of the algorithm should look like and how they should work. Because I still don't have much of a feel for PLP, I went back to the code I'd typed out from the textbook and compared it to the English description I'd written out in an attempt to make sense of what each line actually did and why it was there. At the same time, I tried to run the MIPS code in the PLP tool to see if the errors I got might help me find things I needed to change. I still have no idea what the beginning and end of that code do in terms of using a stack, but at some point a guy in one of my other classes told me I didn't need that part so I cut it out. I also found some differences in the commands like "addi" versus "addiu". These were easy enough to fix.

I ran into trouble when I tried to start optimizing. I replaced the four lines for the swap with just two because I realized I'd already move my values into temporary registers and all I needed to do was store them back reversed. I didn't keep good enough track of my code when I tried to eliminate the "donemaybe" clause. It didn't work and when I put it back, I forgot that I had added an increment line to the outerloop, so I ended up with two which partially broke the thing. It took more than one person reviewing my code to find that error. This taught me to be more careful in tracking my changes. I still would have preferred to know more about PLP going in as I can't imagine how I'd have done if I'd had to start writing with a blank canvas.

As this is a final project, please also reflect on what you learned in this course in general. Please try to be as reflective of your experience as you can. Don't talk about the things you did or the programs themselves, but on your growth from the beginning of the semester to the end. This final part can include growth on the technical aspects or even more general or philosophical growth. Some simple one-sentence examples are, "I finally understood how to learn a new programming language" or "Looking back, I think that the best thing I took away from my interactions with my teammates was how to approach an assignment, and how to get the most out of the experience" (300-400 words spread across 3-5 paragraphs).

I'm going to divide this part into what I learned about myself and my learning process and what I learned about working with other people. I was pretty uncomfortable for much of the time in this class. I felt like I was being assigned tasks without ever being taught the skills I needed to accomplish them. In the "real world" this is generally what things are like so I guess in that way, perhaps this was good and it allowed me to practice this kind of problem solving. I ended up going to great lengths to get the

information I needed including talking to people I wouldn't have approached before and spending a lot of time poring over web pages and the text book. I got better at those things and that's good. It also brought into sharp relief how it feels to not have the tools I needed. It made me want to take a class in PLP. I can only imagine how it must feel to be forced to learn a spoken language only by listening to other people speaking it. It can be done, but it sure takes a lot of work and a lot of time!

I did learn a lot of good technical knowledge and skills. Being forced to write a bubble sort in an unfamiliar language really makes you learn what each little piece does. Of course I also learned a lot about what happens at the level of assembly which enhances your knowledge of the implications of higher level code you write.

I learned a few more things about working with other people in this class. The first is that I'm still not very good at handling people who won't pull their own weight. It's easy to work with a good group but how do you manage a bad one? I tried a few different things but nothing seemed to work. I tried to teach, be patient, be authoritative, micro-manage, be hands off, etc. One team member was going to contribute no matter what I did but the other just wasn't. I'm really not sure if there was anything else to be done.

All in all, I'm going to walk away from this class feeling pretty mixed. I don't understand why it was set up the way it was. I think my thoughts about what undergraduate education should be like have been reinforced. It is possible to learn things in the world just by reading, observing and experimenting. The problem is that it takes a really long time sometimes and doesn't always get you all the skills you need. Taking a class from an expert ideally reduces the time it takes to master new skills because the learning isn't random – it is guided by the expert. That is, at least in part, why you are paying for it. I'm curious to see if my perspective on this changes as time progresses.