# Rebeccah Hunter Final Language Reflection

Just for the record, this has been very difficult to write, because it’s my absolute last thing, and also my last Matt class ever. :[ I keep dragging my feet because after it’s done, that’s it, no more Matt classes, or any Berea CS classes.

1. **What have you learned about interpretation?**

I have learned quite a bit about both parsing and interpretation(thanks Matt, I stayed in Berea this long just for this class, also to graduate, but mostly for this class ;] ). First of all, I now know in depth, the difference between interpretation and compilation. I knew before, but only in a loose sense. It was just a basic thought along the lines of, “Compilers go through the code first, and then run it, and interpreters go through when running line by line.” This basic thought still holds true, but there is much more that I understand now. Parsing is integral to transforming our code into working programs. It takes the high level information and turns it into meaningful information for the interpreter. The interpreter at that point takes that information and actually does the action.

I liked how in the course we used concrete and abstract. Taking the written “concrete” code and abstracting it so it could be interpreted. The use of the terms helped a lot in understanding the roles that parse and interp played in processing the code. I thought when exploring Haskell and Erlang that this information was particularly helpful. Especially in Erlang, which does not have explicitly declared types. Similar in that particular way to python, the language is an interpreted language. Knowing that in both of those languages, the interpreter is what defines those types before the execution of the code was extremely helpful in terms of understanding how to work with the language.

Erlang is also a functional programming language, which I was sort of shaky on before this class. It doesn’t allow variable changes at all, which was really confusing to me, coming from Python. However, working in Racket to implement the tiny interpreter led me to realize that languages can be powerful, regardless of whether they are imperative or not. Working with Erlang was challenging based on the syntax, but because I understood the fundamental principles of what made it function, it was much easier to explore than the first time we explored a language. I would say that Haskell was easier to explore, but I knew much more by the time Ashley and I did our Erlang project, and therefore it was much better.

I also have to take a little time to talk about recursion even though it wasn’t written as part of the prompt. I learned recursion when all Berea students do, in our early core courses. However, I had such a limited understanding, appreciation, and imagination when it came to recursion. Working through list recursion in Racket was integral to me understanding our interpreter. Additionally it helped in terms of understanding efficiency. The code I had written up to this point was almost childish in a way (not trying to be insulting, I just want to communicate that I am much more enlightened now). My previous solutions were functioning code of course, but nowhere near as elegant or efficient as it could have been.

I think the time we took to cover lists and recursion in racket was essential to understanding the principles of the interpreter, but more than that, essential to creating it. If given the task of writing an interpreter before taking this class, I would have no idea where to start, let alone how it even fit in with what I was doing. Now, however, I not only understand where to begin, I think I could do it on my own again, maybe eventually on a larger scale too.

Finally, I should also talk briefly about Lisp. Seeing as I have now been enlightened by the glory that is Racket.. I now know so much more and can spread the true word to all. I would say jokes aside, but I’m not joking about this… Non-jokes aside, I am glad we learned Racket, since it’s based on Scheme/Lisp. I think the prevalence of Lisp being basically the first “high level” language is really important for relating the high level languages to the “low level” information we learn in other courses such as Computer Organization. Lisp was literally what created the first interpreters, so we should create them in Lisp too. We should do that for several reasons, 1. History is cool, 2. It makes sense because it does bridge that gap between things we know like Python and the lower level of computation, creating a better overall understanding of computing, as well as giving us specified and applicable knowledge to tasks we may complete in the future, and 3. It’s nerdy and cool, and we all need more of that to give us life.

1. **What did you do well?**

I loved this course, so it’s hard to say what I did well, because although I have learned a lot, there is always more that I would like to learn. I felt that I did a few things really well though. This last project with Ashley was great. I wish we could work together more often, because we both care a lot about computer science, and we work really well together. Neither of us are passive. We both have had plenty of experience in projects where a partner does all the work and the other partner doesn’t do anything. I will say though, she goes above and beyond, after we were “finished” with everything, she went through it all on her own, adding finishing touches (also unrelated, but that girl can talk SO fast. During our lightning presentations for the exam day, I was so amazed by her, I cannot talk that fast… I’m glad, because if it weren’t for the both of us working to get through our information really quickly, our presentation would have been too long.)

Okay, back to the topic, I think I did pretty well in our Racket assignments and exploration. I really liked the class, so I wanted to do the work it took to understand everything I worked on subst long after we had shifted focus, to make sure that I understood at least loosely, why we did it. I also went from being super nervous and unsure about our assignments at the beginning of the term, to feeling unafraid to tackle something I didn’t know how to do. I whiteboard-ed things out that I didn’t understand, and asked my peers questions. I also got to the point where I felt comfortable helping others. I volunteered to help people at the start of the term, but I think I wouldn’t have been very good at it. By the end of the term, I was really excited to help other people with their understanding of the course and it assignments.

When I wasn’t too weak, I tried very hard to participate and engage in class. I took notes for a long time in DrRacket, and they helped a lot when working on homework. I also continued to work on our DASLANG after we migrated from subst to env and eventually “finished” the group project. I just kept saving and moving up to the next number. I went up to language 7, I think? Not all of that has the environment work in it because I really wanted to make sure I understood the reason we were shifting from subst to env.

\*\*\*WARNING: lot’s of my Racket code is in German… I would say that it’s Karma, but that’s not it, it’s me.\*\*\*

Overall, I spent quite a bit of time on weekends and outside of class on the topics and class work/assignments, and I really enjoyed it. I explored Haskell and Erlang for class, but also looked at Scala and Prolog before committing to projects in either of those, since I had wanted to explore more. I also did some coding in those, even though it wasn’t an assignment for the class, I just wanted to know more (I know it’s not explicitly part of the class, but the class is the reason I did it). I really wish I had explored Erlang earlier, I think it would have been really relevant to my work for Mario in Parallel & Distributed, that being said, TAing for that class gave me a lot of knowledge and information that I could utilise when working on and presenting about Erlang.

1. **What did you do poorly?**

Subst…… I was so confused about subst. I was confused about subst and the shift from subst to environments. I was confused and I should have asked more questions. I was super sick one class, and before I knew it, subst was gone, and we had moved to env, and I had missed how and why. I wish I had asked more questions. I got that we were moving, and that it was going to be better, but nobody could tell me why. I should have asked Matt to sit down with me and go through it. I think I was embarrassed that I had missed it, even though I was in class. I think looking back it’s so easy to know what I should have done, maybe this means next time I will ask for help every time I need it, not sometimes.

Another thing I wish was different was the middle of term when I was super sick. I talked to Matt, and he knew, and I worked really hard to make up for that, but I still wish that I hadn’t been so sick. There were times that I was in class, but it felt like I was barely even there.

One more thing I wish I had done is from early on in the class. I wasn’t always consistent in giving code to the class when we were supposed to. I did it a few times, and didn’t a couple times. I really valued the work that the other students put in, and I should have put in the same effort. Sometimes I was sick, or forgot, but neither of those reasons are really good enough for me. The point was so that we thought through those problems together. I know I was so excited when we went through my contributions in class and analyzed or solved them. I learned and so did everybody else. I would have liked to participate more fully in those opportunities, because it would only be more good learning.

1. **How many classes have you missed, or been substantially late to?**

I have missed very few class days, besides the ones I missed for job interviews or doctor's appointments. However, I was pretty late pretty frequently because of my medical situation. I tried very hard to make it as close to on time as I could, sometimes I was on time or even early, but sometimes I was late. Mostly only a 5-10 minutes late, but I know there were a few days I was about 20 minutes late. I really struggle with weakness and being able to move in the mornings, and hated being late because I really cared about the class. I tried to communicate when I had been having lots of flare ups, but I know that I should have done a better job communicating those times, as they were far more frequent than I let on. I do feel that if the class had been later that I would have been able to engage more fully during more class periods. I really loved the class, so it was a big internal conflict for me that I was so weak during class periods.

1. **If you had to grade yourself this term, in this course, how would you? Why?**

I would give myself an A-. I think this because there is always room for improvement, but given my medical situation, I think I did as well as I could have. I worked hard on this class, even though it isn’t a necessity for me to graduate (I do think it’s a necessity for me as an individual). Again, that being said, there is always room to improve, regardless of where each individual is at, and I always feel that I can improve.