# hello world!

The best way to introduce Computer Science is with the hello world program, so I wrote one in ten different languages. It's an important part of programmer culture, I think. Some languages are verbose; some languages write like every line of code costs thousands of dollars to produce. Some are decades old and are now hallmarks of learning to code like Java or Python. At least one language in the set chosen is less than three years old. That one is new to everyone.

I said that Computer Science is an open-minded discipline. Computers are weird and seemingly easy tasks can be secretly very difficult. Have fun setting up your computer to run each file (bonus points if you are using a Linux box). Open-mindedness is needed both for solving computing problems and for maintaining a healthy life in a computing world. In undergrad, I tried to avoid "raging" at all cost while working on an assignment. Anger closes the mind and breaks monitors.

I said that Computer Science is iterative. It is. You will be told that you are wrong by your compiler thousands of times. You'll write your program again. If you refuse to rage-quit, it'll run. The open-mindedness to approach a computing problem in the first place and the mental strength to try the problem again are both necessary.

I said that Computer Science is about language. Why not about science? Computer Science departments are sometimes closely tied to Math departments. Why not about Math? Computers are physical devices, so why not about computers-as-engineered-thing? The best answer to what is Computer Science is all of the above, but

I feel that there are many places where an engineering or math or science paradigm is applied to Computer Science and that there are very few places where a language-based approach is explicitly used despite a major paradigm of computation being programming language theory with the major computer model as lambda calculus. I thought it'd be fun to push PLT as the main bias in the curriculum to see how far it can go. I also think that Computer Science relies a bit too much on quantitative reasoning as a metric in education when in practice verbal reasoning is being used as much or more during things like programming.

For this first tutorial, get your computer setup to run these programs. That is an assignment in of itself and doesn't require instruction. Use google, open-mindedness, and willingness to try again. Take a look at the similarities and differences between the programs. There is a history connecting many of them. For example, Java, Scala, Kotlin, Python, and Javascript (hidden in html). There is also a close history between Haskell and Cedille. It would also help to know that Haskell and Cedille are functional programming languages and the rest are considered imperative for the most part. That's the second assignment, looking into and making note of the development of these languages (and googling the words "functional" and "imperative").