hand coding round robin

part I ~ lecture

**1. COMPUTATION AS METAPHOR**

In leading up to this workshop, I have been thinking a lot about computation as metaphor. This is because throughout my education as a programmer I have learned that all computational systems are always and already built on top of metaphor. And because of this, computers require that programming, as we know it today, be a set of extremely abstract processes. So, when I’m coding something, I’m often coding on top of someone else’s code, in someone else’s software, on top of the software in my operating system which is nearly permanently baked into the proprietary chip of my Apple computer.

**2. MARK II**

Computers at their core consist of a complex series of sets of on and off switches. This means that everything we do on our computers today needs to abstract away those complexities in order for us to use them in the way that we do. These abstractions allow us to do things like writing text in a file, without thinking about the how or even if the computer will be able to do this and with speed. It is important to note that since the early days of computing computer time has always been more important than human time. When programmers were writing code for computers the size of rooms the time it took to run the program was really expensive, so people were always writing code that was most efficiently read by the computer itself not by the humans who were making the programs.

**3. DESK**

So back to the idea of computation as metaphor. The most obvious metaphor might be the desktop metaphor. The idea of a file inside a folder on a desktop...and there's a trashcan. I’m always wondering, when you put something in the trash, does it ever really go away? The office metaphor might be one which we may never get away from, at least as long as computers are as personal as they are now. But what is the file really? And what is the folder? I just learned that folders aren't actually anything. They don't take up \*any\* space on your computer. But as we saw in Melanie’s folder poetry workshop, folders and files and the navigation between the two can be a poetic act.

**4. ILLUSION OF POWER**

This is all just to highlight that there has been a significant shift from physical programming, the act of physically adjusting switching and weaving magnetic core to abstracted programming, building software on top of software. Since folders and files, which if we looked closely at them might not resemble anything close to a piece of paper inside another folded piece of paper. I’m left wondering if computation is understood through metaphors does the way that we use our computers also require some kind of illusion of power over our computers.

**5. GRAMMERS**

I think it's important to note this illusion. I think having an illusion of control leads programmers down a potentially dangerous path. Programmers, through the creation of complex processes and algorithmic systems go through so many hurdles trying to get the computer to do what they want it to do. When the computer finally obeys their commands, they are granted with this feeling of immense gratification and power. I’ve felt this feeling of gratification as a programmer after hours and hours of trying to debug something I wrote. It’s a cyclical command and response from you to your computer.

**6. VACUUM**

I'm wondering, if the process of programming a smart device, for instance, is rooted in notions of command and control and illusion and power, and if all computation is metaphor, does this make for dangerous results? Does it perhaps mean that programmers get so caught up in the power dynamic between themselves and their computer that they lose sight of how what they're doing lives in the real world alongside real humans?

**7. WRITING EMAILS**

This is a video of my grandma writing an email. It makes me wonder what it would be like if programmers had to press the keys of their keyboards so gently that it required them to move extremely slow and therefore with care? what would it mean if programmers had to manually move the switches to on and off positions in order to make a letter appear in an code they were writing? Or what would it be like if we could hear all of the computation happening inside our laptops as we typed out an email?

**8. CAMERONS WORLD OF GEOCITIES**

And so, all of these questions have led me to care deeply about hand coding. Hand coding is the process of coding that bridges the gap between you as the programmer and your computer as the programmed. In other words, it is a way to code which is not dependent on someone else’s software or library, instead it is the process of writing in the inherent language of the browser. Your browser is software which translates HTML, CSS and JavaScript into web pages. More specifically your browser, as the performer of the program you are writing. The code you will write in this workshop will be very particular and you will write out each character slowly and by hand, no copying or pasting. You will also do this on each other computers. The intention here is to perhaps erode this feeling of ultimate control that you have over your computer's system and to move carefully (but not too carefully) around someone else's.