Danny Theisen

Dr. Alexander

Honors 190

10/30/12

A Brief History of Computing

Arguably one of the greatest inventions of the 21st century is the personal computer. Computers saturate our lives and have changed almost every aspect of our day-to-day activities. But for something so common, it’s amazing how little most people know about them. Pick up your phone and look at it. What do you really know about how it works? You feed your phone some electricity, the black boxes inside it swap some data, and then, bam, like magic they make your new high-score in angry birds flash across the top of the screen. Sounds about right, yeah? Well, what do you know about how your phone is made? How about who invented all those black boxes and digital magic tricks? For being considered one of the man’s greatest inventions, it is amazing how little most of us really know about computers. As long as the electrons in our phone continue to form funny pictures of cats from the internet, we don’t give a second thought to how it functions, where it came from, or most importantly, who invented it. We ignore how all that data was transformed into knowledge.   
 Fortunately, the invention of computing is exactly what our honors seminar is about. Almost everyone knows how Henry Ford revolutionized the automotive industry, but how many people know how Alan Turing helped conceptualize the computing industry? Like Henry Ford, Turing is also considered the father of his craft; however, unlike Ford, Turing’s contributions to his industry are a little more complicated than choosing to rearrange some workers. To put it simply, Turing taught the math world what they are trying to teach us in the Honors program: how to convert data into knowledge. For example, one of his earliest feats was creating a machine that could decrypt enemy German messages during WWII. Nowadays that doesn’t sound too impressive, but back before we had computers (or even calculators for that matter), it was an astounding feat. Turing took a new statistical approach to cryptography and used it to create an electro-mechanical machine he called The Bombe. This machine would rapidly cycle through potential ciphers the German’s might have used and then rule out ciphers that produced contradictions. It sounds simple, but transforming mathematical logic into a physical machine was revolutionary for the time, and ultimately proved to the world the value of learning to analytically define problems in terms of logic. Turing perpetuated this idea of using mathematical logic for problem solving throughout his life and later spent much of his time aiding in early computer design and discussing Artificial Intelligence. However, Turing wasn’t the only person who was working on mathematical logic during his time, and even prior to his work on the Bombes, two mathematicians were forming the basis for the system of logic that Turing would later apply. The first of these men was David Hilbert, a mathematician who would be any math student’s nightmare. Hilbert is famous for making some of the most difficult math problems in history and is most renowned for his system of 23 problems that he claimed was the key to the future of mathematics. These problems were created in 1900, and while some of them were solved quickly, there are still a handful that our best mathematicians continue to struggle with today. However, in terms of computer science, the most important of these problems was Hilbert’s 10th problem. In essence, what it stated was “Is there a way to determine whether a mathematical proposition could be proven,” or in other words, is there a way to tell that a mathematical hypothesis can be proven before you try to prove it. Why is this important to computing? Well, if scientists had been able to prove the question true, then no one would ever have had to suffer the agony of deciding whether to reboot your computer or wait just a few more minutes to see if a program will ever unfreeze. If we could prove something had an answer before we tested it, then we would know if computer programs would freeze before we even ran them. Unfortunately, this is where our second mathematician comes in, the problem solver, Kurt Godel. Godel proposed an idea called the incompleteness theorem, which stated that if a system of axioms is consistent, then it cannot prove that it is consistent with its own axioms. This theorem, along with help from Alan Turing’s concept of a Turing Machine, eventually helped prove Hilbert’s 10th problem to be false; and while disappointing, ultimately forced computer science to stop trying to take shortcuts, but instead made scientists buckle down and design a more formalized system of logic.

See? The founders of computer science are not so bad. Just a little… complicated. Understanding Henry Ford’s assembly line is easy, but in the 21st century it’s more important for us to understand where the hundreds of computers we interact with came from. So next time someone asks you for a famous thinker, give Turing a chance and prove why he was one of the masters of turning data into knowledge.