



Early Programming: Crash Course Computer Science #10

Crash Course: Computer Science

<https://youtube.com/watch?v=nwDq4adJwzM>

<https://nerdfighteria.info/v/nwDq4adJwzM>

===== (00:00) to (02:00) =====

(PBS Digital Intro)

Hi, I'm Carrie Anne, and welcome to CrashCourse: Computer Science. Over the last few episodes, we've talked a lot about the mechanics of how computers work, how they use complex circuits to save and retrieve values from memory and perform operations on those values, like adding two numbers together. We've even briefly talked about sequences of operations, which is a computer program. What we haven't talked about is how a program gets into a computer. You might remember in episode seven and eight, when we stepped through some simple example programs through the CPU that we had created. For simplicity, we just waved our hands and said that the program was already magically in memory, but in reality, programs have to be loaded into a computer's memory. It's not magic, it's computer science.

(Intro)

The need to program machines existed way before the development of computers. The most famous example of this was in textile manufacturing. If you just wanted to weave a big red tablecloth, you could simply feed red thread into a loom and let it run. What about if you wanted the cloth to have a pattern like stripes or plaid? Workers would have to periodically reconfigure the loom as dictated by the pattern. But this was labor intensive, which made pattern fabrics expensive.

In response, Joseph Marie Jacquard developed a programmable textile loom, which he first demonstrated in 1801. The pattern for each row of the cloth was defined by a punched card, with the presence or absence of a hole in the card determined if a specific thread was held high or low in the loom. Such as the cross-thread, called the weft, passed above or below the thread. To vary the pattern across rows, these punch cards were arranged in long chains, forming a sequence of commands for the loom. Sound familiar? Many consider Jacquard's loom to be one of the earliest forms of programming.

Punched cards turned out to be a cheap, reliable, and fairly human readable way to store data. Merely a century later, punched cards were used to help tabulate the 1890 US Census, which we talked about in episode one. Each card held an individual person's data, things like race, marital status, number of children, country of birth, and so on. For each demographic question, a census worker would punch out a hole of the appropriate position.

===== (02:00) to (04:00) =====

When a card was fed into the tabulating machine, a hole would cause the running total for that specific answer to be increased by one. In this way, you could feed the entire county's worth of people and at the end, you'd have running totals for all of the questions that you asked.

It's important to note here that early tabulating machines were not truly computers. They could only do one thing: tabulate. Their operation was fixed and not programmable. Punch cards stored data, but not a program. Over the next sixty years, these business machines grew in capability, adding features to subtract, multiply, divide, and even make simply decisions about when to perform certain operations.

To trigger these functions appropriately so that different calculations could be performed, a programmer accessed a control panel. This panel was full of little sockets, into which a programmer would plug cables to pass values and signals between different parts of the machine. For this reason, they were also called

'plugboards'. Unfortunately, this meant having to rewire the machine each time a different program needed to be run, and so by the 1920s, these plugboards were made swappable. This not only made programming a lot more comfortable, but also allowed for different programs being plugged into a machine. For example, one board might be wired to calculate sales tax, while another helped with payroll, but plugboard were fiendishly complicated to program. This tangle of wires is a program for calculating a profit-loss summary, using an IBM 402 accounting machine, which were popular in the 1940s, and this style of plugboard programming wasn't unique through electromechanical computers.

The world's first general purpose electronic computer, the ENIAC completed in 1946, used a ton of them. Even after a program had been completely figured out on paper, physically wiring up the ENIAC and getting the program to run could take upwards of three weeks. Given the enormous cost of these early computers, weeks of downtime simply to switch programs was unacceptable and a new, faster, more flexible way to program machines was badly needed.

Fortunately, by the late 1940s and into the 50s, electronic memory was becoming feasible. As cost fell, memory size grew. Instead of storing a program as a physical plugboard of wires, it became possible to store a program entirely in a computer's memory, where it could be easily changed by programmers and quickly accessed by the CPU.

===== (04:00) to (06:00) =====

These machines were called stored program computers. With enough computer memory, you could store not only the program you wanted to run, but also any data your program would need, including new values it created along the way. Unifying the program and data into a single shared memory is called the Von Neumann Architecture, named after John Von Neumann, a prominent mathematician-physicist who worked on the Manhattan project and several early electronic computers and once said, "I am thinking about something much more important than bombs. I am thinking about computers."

The hallmarks of a Von Neumann computer are a processing unit containing an arithmetic logic unit, data registers, an instructional register, and instruction address register. But finally, a memory to store both data and instructions. Hopefully, this sounds familiar, 'cause we actually built a Von Neumann computer in episode seven.

The very first Von Neumann Architecture stored program computer was constructed in 1948, by the University of Manchester, nicknamed 'Baby', and even the computer you're watching this video on right now uses the same architecture. Now, electronic computer memory is great and all, but you still have to load the program and data into the computer before it can run, and for this reason, punch cards were used. Let's go to the Thought Bubble.

Well into the 1980s, almost all computers had a punch card reader, which could suck in a single punch card at a time and write the contents of the card into the computer's memory. If you loaded in a stack of punch cards, the reader would load them all into memory sequentially, as a big block. Once the program and data were in memory, the computer would be told to execute it. Of course, even simple computer programs might have hundreds of instructions, which meant the programs were stored as stacks of punch cards. So if you ever had the misfortune of accidentally dropping your program on the floor, it could take you hours, days, or even weeks to put the code back in the right order.

A common trick was to draw a diagonal line on the side of the card



Early Programming: Crash Course Computer Science #10

Crash Course: Computer Science

<https://youtube.com/watch?v=nwDq4adJwzM>

<https://nerdfighteria.info/v/nwDq4adJwzM>

stack called 'striping', so you'd have at least some clue how to get it back into the right order. Phew! The largest program ever punched into punch cards was the US Air Force's SAGE air defense system completed in 1955. At its peak, the project is said to have employed 20% of the world's programmers!

Chad & Stacey Emigholz studio in Indianapolis, Indiana, and it was made with the help of all these nice people and our wonderful graphics team, ThoughtCafe. Thanks to the random access memories, I'll see you next time.

===== (06:00) to (08:00) =====

Its main control program was stored on a whopping 62,500 punch cards, which is equivalent to roughly 5mb of data. Pretty underwhelming by today's standards, and punch cards weren't only used for getting data into computers, but also getting data out of them. At the end of a program, results could be written out of computer memory and on to a punch card by, well, punching cards. Then, this data could be analyzed by humans or loaded into a second program for additional computation. Thanks, Thought Bubble.

A close cousin to punch cards was punched paper tape, which was basically the same idea, but continuous instead of being on individual cards, and of course, we haven't talked about hard drives, CD-ROMs, DVDs, USB thumb drives, and other similar goodies. We'll get to those more advanced types of data storage in a future episode.

Finally, in addition to plugboards and punch paper, there was another kind of way to program and control computers pre-1980: Panel programming. Rather than having to physically plug in cables to activate certain functions, this could also be done with huge panels full of switches and buttons, and there were indicator lights to display the status of various functions of values and memories. Computers of the 50s and 60s often featured huge control consoles that looked like this. Although it was rare to input a whole program using just switches, it was possible, and early home computers made for hobbyist market used switches extensively, because most home users couldn't afford expensive peripherals like punch card readers.

The first commercially successful home computer was the ALTAIR 8800 were sold in two versions: preassembled and as a kit. The kit, which was popular with amateur computing enthusiasts, sold from the then-unprecedented low price of around \$400 in 1975, or about \$2000 in 2017. To program the 8800, you'd literally toggle the switches on the front panel to enter the binary opcodes and instructions you wanted. Then, you'd press the deposit button to write that value into memory. Then, in the next location in memory, you'd toggle the switches again for your next instruction, deposit it, and so on.

===== (08:00) to (09:27) =====

When you'd finally entered your whole program into memory, you would toggle the switches to be back to memory address zero, press the run button, and watch the little lights blink. That was home computing in 1975, wow. Whether it was plugboards, switches, or punch paper, programming these early computers was the realm of experts. By the professionals who did this for a living or technology enthusiasts, you needed intimate knowledge of the underlying hardware, so things like process opcodes and register (?~[8:29](#)) to write programs. This meant programming was hard and tedious, and even professional engineers and scientists struggled to take full advantage of what computers could offer. What was needed was a simpler way to tell computers what to do, a simpler way to write programs, and that brings us to programming languages, which we'll talk about next episode. See you next week.

CrashCourse: Computer Science is produced in association with PBS Digital Studios. At their channel, you can check out a playlist of shows like Brain Craft, (?~[8:58](#)). This episode was filmed at the