



Keyboards & Command Line Interfaces: Crash Course Computer Science #22

Crash Course: Computer Science

<https://youtube.com/watch?v=4RPtJ9UyHS0>

<https://nerdfighteria.info/v/4RPtJ9UyHS0>

Hi, I'm Carrie Anne, and welcome to CrashCourse Computer Science! We've talked a lot about inputs and outputs in this series, but they've mostly been between different parts of a computer – like outputting data from RAM or inputting instructions to a CPU. We haven't discussed much about inputs coming from humans. We also haven't learned how people get information out of a computer, other than by printing or punching it onto paper. Of course, there's a wide variety of input and output devices that allow us users to communicate with computers. They provide an interface between human and computer and today, there's a whole field of study called Human-Computer Interaction. These interfaces are so fundamental to the user experience that they're the focus of the next few episodes.

INTRO

As we discussed at the very beginning of the series, the earliest mechanical and electro-mechanical computing devices used physical controls for inputs and outputs, like gears, knobs and switches, and this was pretty much the extent of the human interface. Even the first electronic computers, like Colossus and ENIAC, were configured using huge panels of mechanical controls and patch wires. It could take weeks to enter in a single program, let alone run it, and to get data out after running a program, results were most often printed to paper. Paper printers were so useful that even Babbage designed one for his Difference Engine, and that was in the 1820s! However, by the 1950s, mechanical inputs were rendered obsolete by programs and data stored entirely on mediums like punch cards and magnetic tape. Paper printouts were still used for the final output, and huge banks of indicator lights were developed to provide real time feedback while the program was in progress.

It's important to recognize that computer input of this era was designed to be as simple and robust as possible for computers. Ease and understanding for users was a secondary concern. Punch tape is a great example – this was explicitly designed to be easy for computers to read. The continuous nature of tape made it easy to handle mechanically, and the holes could be reliably detected with a mechanical or optical system, which encoded instructions and data. But of course, humans don't think in terms of little punched holes on strips of paper. So, the burden was on programmers. They had to spend the extra time and effort to convert their ideas and programs into a language and a format that was easy for computers of the era to understand – often with the help of additional staff and auxiliary devices.

It's also important to note that early computers, basically pre-1950, had an extremely simple notion of human input. Yes, humans input programs and data into computers, but these machines generally didn't respond interactively to humans. Once a program was started, it typically ran until it was finished. That's because these machines were way too expensive to be waiting around for humans to type a command or enter data. Any input needed for a computation was fed in at the same time as the program.

This started to change in the late 1950s. On one hand, smaller-scale computers started to become cheap enough that it was feasible to have a human-in-the loop; that is, a back and forth between human and computer. And on the other hand, big fancy computers became fast and sophisticated enough to support many programs and users at once, what were called multitasking and time-sharing systems. But these computers needed a way to get input from users. For this, computers borrowed the ubiquitous data entry mechanism of the era: keyboards.

At this point, typing machines had already been in use for a few centuries, but it was Christopher Latham Sholes, who invented the modern typewriter in 1868. It took until 1874 to refine the design

and manufacture it, but it went on to be a commercial success. Sholes' typewriter adopted an unusual keyboard layout that you know well – QWERTY – named for the top-left row of letter keys. There has been a lot of speculation as to why this design was used. The most prevalent theory is that it put common letter pairings in English far apart to reduce the likelihood of type bars jamming when entered in sequence. It's a convenient explanation, but it's also probably false, or at least not the full story. In fact, QWERTY puts many common letter pairs together, like "TH" and "ER". And we know that Sholes and his team went through many iterations before arriving at this iconic arrangement. Regardless of the reason, the commercial success of Sholes' typewriter meant the competitor companies that soon followed duplicated his design. Many alternative keyboard layouts have been proposed over the last century, claiming various benefits. But, once people had invested the time to learn QWERTY, they just didn't want to learn something new. This is what economists would call a switching barrier or switching cost. And it's for this very basic human reason that we still use QWERTY keyboards almost a century and a half later!

I should mention that QWERTY isn't universal. There are many international variants, like the French AZERTY layout, or the QWERTZ layout common in central Europe. Interestingly, Sholes didn't envision that typing would ever be faster than handwriting, which is around 20 words per minute. Typewriters were introduced chiefly for legibility and standardization of documents, not speed. However, as they became standard equipment in offices, the desire for speedy typing grew, and there were two big advances that unlocked typing's true potential.

Around 1880, Elizabeth Longley, a teacher at the Cincinnati Shorthand and Type-Writer Institute, started to promote ten-finger typing. This required much less finger movement than hunt-and-peck, so it offered enhanced typing speeds. Then, a few years later, Frank Edward McGurrian, a federal court clerk in Salt Lake City, taught himself to touch-type; as in, he didn't need to look at the keys while typing. In 1888, McGurrian won a highly publicized typing-speed contest, after which ten-finger, touch-typing began to catch on. Professional typists were soon able to achieve speeds upwards of 100 words per minute, much faster than handwriting! And nice and neat too!

So, humans are pretty good with typewriters, but we can't just plunk down a typewriter in front of a computer and have it type – they have no fingers! Instead, early computers adapted a special type of typewriter that was used for telegraphs, called a teletype machine. These were electro-mechanically-augmented typewriters that could send and receive text over telegraph lines. Pressing a letter on one teletype keyboard would cause a signal to be sent, over telegraph wires, to a teletype machine on the other end, which would then electro-mechanically type that letter. This allowed two humans to type to one another over long distances... basically a steampunk version of a chat room. Since these teletype machines already had an electronic interface, they were easily adapted for computer use, and teletype computer interfaces were common in the 1960s and 70s.

Interaction was pretty straightforward. Users would type a command, hit enter, and then the computer would type back. This text "conversation" between a user and a computer went back and forth. These were called command line interfaces, and they remained the most prevalent form of human-computer interaction up until around the 1980s. Command Line interaction on a teletype machine looks something like this. A user can type any number of possible commands. Let's check out a few, beginning with seeing all of the files in the current directory we're in. For this, we would type the command, "ls", which is short for list, and the computer replies with a list of the files in our current directory. If we want to see what's in our "secret Star Trek Discovery Cast dot t-x-t file",



Keyboards & Command Line Interfaces: Crash Course Computer Science #22

Crash Course: Computer Science

<https://youtube.com/watch?v=4RPtJ9UyHS0>

<https://nerdfighteria.info/v/4RPtJ9UyHS0>

we use yet another command to display the contents. In UNIX, we can call “cat” - short for concatenate. We need to specify which file to display, so we include that after the command, called an argument. If you’re connected to a network with other users, you can use a primitive version of a Find My Friends app to get more info on them with the command “finger”.

Electromechanical teletype machines were the primary computing interface for most users up until around the 1970s. Although computer screens first emerged in the 1950s, and were used for graphics they were too expensive and low resolution for everyday use. However, mass production of televisions for the consumer market, and general improvements in processors and memory, meant that by 1970, it was economically viable to replace electromechanical teletype machines with screen-based equivalents. But, rather than build a whole new standard to interface computers with these screens, engineers simply recycled the existing text-only, teletype protocol.

These machines used a screen, which simulated endless paper. It was text in and text out, nothing more. The protocol was identical, so computers couldn’t even tell if it was paper or a screen. These virtual teletype or glass teletype machines became known as terminals. By 1971, it was estimated, in the United States, there was something on the order of 70,000 electro-mechanical teletype machines and 70,000 screen-based terminals in use. Screens were so much better, faster and more flexible, though. Like, you could delete a mistake and it would disappear. So, by the end of the 1970s, screens were standard.

You might think that command line interfaces are way too primitive to do anything interesting. But even when the only interaction was through text, programmers found a way to make it fun. Early interactive, text-based computer games include famous titles like Zork, created in 1977. Players of these sorts of early games were expected to engage their limitless imaginations as they visualized the fictional world around them, like what terrifying monster confronted them when it was pitch black and you were likely to be eaten by a grue.

Let’s go back to our command line, now on a fancy screen-based terminal, and play! Just like before, we can see what’s in our current directory with the “ls” command. Then, let’s go into our games directory by using the “cd” command, for “change directory”. Now, we can use our “ls” command again to see what games are installed on our computer. Sweet, we have Adventure! All we have to do to run this program is type its name. Until this application halts, or we quit it, it takes over the command line. What you’re seeing here is actual interaction from “Colossal Cave Adventure”, first developed by Will Crowther in 1976. In the game, players can type in one- or two-word commands to move around, interact with objects, pickup items and so on. The program acts as the narrator, describing locations, possible actions, and the results of those actions. Certain ones resulted in death! The original version only had 66 locations to explore, but it’s widely considered to be the first example of interactive fiction.

These text adventure games later became multiplayer, called MUDs or Multi-User Dungeons. And they’re the great-forebearers of the awesome graphical MMORPG’s (massive, multiplayer online role playing games) we enjoy today. And if you want to know more about the history of these and other games we’ve got a whole series on it hosted by Andre Meadows!

Command line interfaces, while simple, are very powerful. Computer programming is still very much a written task, and as such, command lines are a natural interface. For this reason, even today, most programmers use command line interfaces as part of their work. And they’re also the most common way to access

computers that are far away, like a server in a different country. If you’re running Windows, macOS or Linux, your computer has a command line interface – one you may have never used. Check it out by typing “cmd” in your Windows search bar, or search for Terminal on Mac. Then install a copy of Zork and play on!

So, you can see how these early advancements still have an impact on computing today. Just imagine if your phone didn’t have a good ol’ fashioned QWERTY keyboard. It could take forever to type your Instagram captions. But, there’s still something missing from our discussion. All the sweet sweet graphics! That’s our topic for next week. See you soon.

Hey guys, before we go I wanted to tell you about a new show produced by Complexly and PBS Digital Studios called Eons. Eons is about exploring the history of life on earth all the way up to the most recent ice age and yes, dinosaurs. Check out the link in the description and subscribe. We hope you like it as much as we do.

Crash Course Computer Science is produced in association with PBS Digital Studios. At their channel you can check out a playlist of shows like Brain Craft, Coma Niddy, and PBS Infinite Series. This episode was filmed at the Chad and Stacy Emigholz Studio in Indianapolis, Indiana and it was made with the help of all of these nice people and our wonderful graphics team Thought Cafe. Thanks for the random access memories, I’ll see you next time.