
Contents

Preface	xi
Introduction	1
Part I: Hardware	7
1. What Is a Computer?	11
1.1 Logical Construction	13
1.1.1 Processor	14
1.1.2 Primary memory	15
1.1.3 Secondary storage	16
1.1.4 Et cetera	18
1.2 Physical Construction	18
1.3 Moore's Law	22
1.4 Summary	24
2. Bits, Bytes, and Representation of Information	27
2.1 Analog versus Digital	27
2.2 Analog-Digital Conversion	29
2.2.1 Digitizing images	29
2.2.2 Digitizing sound	30
2.2.3 Digitizing movies	33
2.2.4 Digitizing text	33
2.3 Bits, Bytes, and Binary	34
2.3.1 Bits	35
2.3.2 Powers of two and powers of ten	36
2.3.3 Binary numbers	37
2.3.4 Bytes	39
2.4 Summary	41
3. Inside the Processor	43
3.1 The Toy Computer	44
3.1.1 The first Toy program	44
3.1.2 The second Toy program	46

3.1.3 Branch instructions	47
3.1.4 Representation in memory	49
3.2 Real Processors	50
3.3 Caching	53
3.4 Other Kinds of Computers	54
3.5 Summary	56
Wrap-up on Hardware	57
Part II: Software	59
4. Algorithms	63
4.1 Linear Algorithms	64
4.2 Binary Search	66
4.3 Sorting	67
4.4 Hard Problems and Complexity	71
4.5 Summary	74
5. Programming and Programming Languages	75
5.1 Assembly Language	76
5.2 High-Level Languages	77
5.3 Software Development	84
5.3.1 Libraries, interfaces, and development kits	85
5.3.2 Bugs	86
5.4 Intellectual Property	89
5.4.1 Trade secrets	89
5.4.2 Trademarks	89
5.4.3 Copyrights	89
5.4.4 Patents	91
5.4.5 Licenses	92
5.5 Standards	94
5.6 Open Source Software	95
5.7 Summary	97
6. Software Systems	99
6.1 Operating Systems	100
6.2 How an Operating System Works	105
6.2.1 System calls	106
6.2.2 Device drivers	107
6.3 Other Operating Systems	107
6.4 File Systems	108
6.4.1 Secondary storage file systems	110
6.4.2 Removing files	112
6.4.3 Other file systems	113
6.5 Applications	115
6.6 Layers of Software	117
6.7 Summary	118
7. Learning to Program	121
7.1 Programming Language Concepts	122
7.2 A First JavaScript Program	123
7.3 A Second JavaScript Program	124
7.4 Loops and Conditionals	126

CONTENTS

ix

7.5	JavaScript Libraries and Interfaces	128
7.6	How JavaScript Works	129
7.7	A First Python Program	130
7.8	A Second Python Program	131
7.9	Python Libraries and Interfaces	133
7.10	How Python Works	135
7.11	Programming with AI	136
7.12	Summary	137
Wrap-up on Software		139
Part III: Communications		141
8. Networks		147
8.1	Home Networking	148
8.2	Local Area Networks and Ethernet	149
8.3	Wireless	151
8.4	Cell Phones	153
8.5	Bandwidth	157
8.6	Compression	158
8.7	Error Detection and Correction	160
8.8	Summary	162
9. The Internet		163
9.1	An Internet Overview	164
9.2	Domain Names and Addresses	167
9.2.1	Domain Name System	167
9.2.2	IP addresses	168
9.2.3	Root servers	169
9.2.4	Registering your own domain	170
9.3	Routing	171
9.4	TCP/IP Protocols	174
9.4.1	IP, the Internet Protocol	174
9.4.2	TCP, the Transmission Control Protocol	175
9.5	Higher-Level Protocols	177
9.5.1	Telnet and SSH: remote login	177
9.5.2	SMTP: Simple Mail Transfer Protocol	178
9.5.3	File sharing and peer-to-peer protocols	180
9.6	Copyright on the Internet	181
9.7	The Internet of Things	183
9.8	Summary	184
10. The World Wide Web		187
10.1	How the Web Works	188
10.2	HTML	189
10.3	Cookies	192
10.4	Active Content in Web Pages	193
10.5	Active Content Elsewhere	194
10.6	Viruses, Worms and Trojan Horses	195
10.7	Web Security	197
10.7.1	Attacks on clients	197
10.7.2	Attacks on servers	202

10.7.3 Attacks on information in transit	203
10.8 Defending Yourself	204
10.9 Summary	207
Part IV: Data	209
11. Data and Information	211
11.1 Search	212
11.2 Tracking	215
11.3 Social Media	222
11.4 Data Mining and Aggregation	225
11.5 Cloud Computing	227
11.6 Summary	231
12. Artificial Intelligence and Machine Learning	233
12.1 Historical Background	234
12.2 Classical Machine Learning	236
12.3 Neural Networks and Deep Learning	237
12.4 Natural Language Processing	240
12.5 Large Language Models	241
12.5.1 Training	242
12.5.2 Transformers and Attention	243
12.5.3 Generating output	243
12.5.4 Limitations	244
12.5.5 The Future of LLMs	244
12.6 Summary	245
13. Privacy and Security	249
13.1 Cryptography Background	250
13.2 Secret-key Cryptography	252
13.3 Public-key Cryptography	253
13.4 Public-key Crypto in Use	254
13.5 Anonymity	258
13.5.1 Tor and the Tor Browser	259
13.5.2 Bitcoin	261
13.6 Summary	263
14. What Comes Next?	267
Notes	273
Glossary	281
Index	289

Introduction

“I fear none of the existing machines; what I fear is the extraordinary rapidity with which they are becoming something very different to what they are at present.”

Samuel Butler, *Erewhon*, Chapter XXIII, 1872.

The computing world changed suddenly on November 30, 2022. On that day, OpenAI, a company that few other than technology enthusiasts had even heard of, released a program called ChatGPT that within a couple of months had 100 million users, and today has a billion.

ChatGPT could carry on a sensible conversation in perfect English and other languages as well. It could answer questions, summarize documents, synthesize prose and poetry, all with remarkable fluency and naturalness.

As is often the case in the education business, I learned about ChatGPT from an undergraduate friend, who had been in my class a couple of years earlier. His first thought was to try ChatGPT on my midterm exam. For the record, it did a creditable job, better than at least some members of the class. ChatGPT wasn’t great at arithmetic, but then again, that has been true of a discouraging number of students over the years. It sometimes hallucinated, or invented things that showed that its knowledge was imperfect. Of course that was true of students as well. Its prose was sometimes a bit pedestrian or predictable, as when I asked it to write letters to admissions officers about why it wanted to attend Princeton and (separately) Harvard, but there were no grammar or spelling errors. Again, this was better than at least some students I have known.

Even then it was an amazing experience to play with it. Now, after several years of evolution, chatbots like ChatGPT, Claude, Gemini, and a dozen others have become staples of daily life for many people. They are more capable in more areas. Many of the early deficiencies are much reduced—fewer hallucinations, better arithmetic, more sophisticated and polished prose. They still get things wrong or make them up, but less frequently.

They can handle much longer inputs. They also do a fine job with different kinds of input, like scanning PDF documents, finding the tables of data within, and converting those to spreadsheet format. They draw pictures and make diagrams. They can carry out extended chains of reasoning, and explain how they are doing it. They now include citations to sources from which they are drawing information. They can even perform actions under some circumstances, like controlling a browser. “Ironic” doesn’t begin to capture what’s going on when a chatbot presses the “I’m a human” checkbox on a web page.

We are in the early stages of what promises to be a revolution in how we work with computers. Some fields have already seen significant effects, and others are sure to. Among the potentially threatened jobs might be legal research (though courts and lawyers have become exquisitely sensitive to the problem of hallucinated legal citations). People who make low-quality infographics or article summaries are likely to become unemployed. Those who translate between major natural languages may be at risk, though not yet for niche ones nor for high-quality literary material. Conspiracy theorists and the creators of fake news, fake images, fake videos and other kinds of misinformation can be replaced by programs that will do it as well but in much greater quantity.

On the search engine front, many people now search with a chatbot rather than with Google or Bing. It’s easier to ask a complicated question or a sequence of them, and the results are likely to be better. Unfortunately, advertisements, clickbait and other junk have arrived or are on the horizon for AI search, and the distinction between AI and conventional search is blurring.

In my own area, computer programming, it’s clear that certain kinds of programming can be done just fine by AI systems. This includes the assignments in introductory programming courses, so naturally students are going to submit solutions that they didn’t write themselves. At the moment, teachers can usually detect that kind of cheating, sometimes because the AI systems are more sophisticated and knowledgeable than typical students would be. Some kinds of entry-level programming can be done sufficiently well by an AI assistant that there is less need for entry-level programmers. That’s likely to be of concern to students looking for jobs and to their instructors, who worry about how to future-proof what we teach them.

Professional programmers use chatbots to assist with their own work; the programs can generate boilerplate code, look up documentation, generate tests, and perform similar tasks quite well. Discussion forums like Hacker News that are frequented by programmers are of several minds on how effective AI assistance is, but on balance more people find it helpful than don’t, and my bet is that AI assistance for programmers will become standard operating procedure.

In short, we’re in the early stages of a massive change in our relationships with computing. Technology has already become deeply embedded in our lives, and although it has improved them in many ways, there’s another side to the story, not so optimistic.

The Internet was already a hotbed of paranoia, hatred and crackpot theories before and during Covid. Today social media has enabled politicians and government officials to spread lies, divide us further, and avoid blame, abetted by “news” outlets with no regard for facts. Sites like Twitter (now called X) and Facebook (now Meta)

tried unsuccessfully to find middle ground between being neutral platforms for the free expression of ideas and limiting the barrage of incendiary postings and outright falsehoods. For the most part, they have given up the attempt; money is winning out over a moral position. The technology billionaires who control such sites are not helping.

Surveillance has reached new heights, used in many countries to restrict people and monitor and control their behavior. For instance, China uses face recognition for, among other things, keeping track of its minority populations. During the Covid-19 pandemic, the Chinese government mandated installation of an app that worked as a sort of immunity passport but also reported its users' locations to the police. In the US and the UK, local law enforcement agencies use face recognition, license plate readers, and the like to keep an eye on people. In the US, the Immigration and Customs Enforcement (ICE) agency is using data from many sources in pogrom-like sweeps that round up potentially illegal immigrants.

Our mobile phones continuously monitor our locations and numerous companies can aggregate the data. Tracking applications for smartphones are an excellent example of the double-sided nature of technology. Who could have been against a Covid-19 contact-tracing system that tells you whether you have been exposed to a potentially contagious person? But any technology that enables the government to know where you've been and who you've been talking to also helps it to monitor and control more effectively. It's a short and slippery slope from disease tracking to ferreting out peaceful protesters, dissidents, political enemies, whistle-blowers, and anyone else that the authorities think might be a threat to their power.

For almost all of our interactions with the online world and in the real world as well, countless computer systems watch and remember who we dealt with, how much we paid, and where we were at the time. A large part of this data gathering is for commercial purposes, since the more that businesses know about us, the more accurately they can target us for advertising. Most readers know that such data is collected, but I expect that many would be surprised by how much there is and how detailed. In 2014, Shoshana Zuboff coined the term *surveillance capitalism* to describe how businesses collect and use personal information on a grand scale. It's disturbingly accurate.

Companies are not the only observers; governments are deeply involved in surveillance as well. The NSA emails, internal reports, and PowerPoint presentations disclosed by Edward Snowden revealed much about spying in the digital era. The gist is that the NSA watches everyone on a grand scale.

Snowden's revelations were stunning. It had been widely suspected that the NSA spied on more people than it admitted, but the extent surpassed everyone's imagination. The NSA routinely collected metadata about all telephone calls made in the US—who called who, when they talked, and for how long—and may have recorded the content of these calls as well. It recorded my Skype conversations and email contacts, and probably the mail contents as well. (Yours too, of course.) It tapped the cell phones of world leaders. It intercepted huge amounts of Internet traffic by placing recording devices on equipment where submarine cables enter and leave the US. It enlisted or coerced the major telecommunications and Internet companies to gather and hand over information about their users. It stored great amounts of data for

extended periods of time, sharing some of it with spy agencies in other countries.

Meanwhile, back on the commercial front, hardly a day goes by when we don't learn of another breach of security at some company or institution, in which shadowy hacker groups steal names, addresses, credit card numbers, and other personal information about millions of people. Usually these are high-tech criminals, but sometimes it's espionage by other countries, looking for valuable information. From time to time, foolish or careless behavior by whoever maintains the information accidentally exposes private data. No matter what the mechanism, data that has been collected about us is all too often exposed or stolen, potentially to be used against us.

The purpose of this book is to explain the technology—hardware, software, communications—that lies behind all of this, so you understand how such systems operate. How can pictures, music, movies, and intimate details of your personal life be sent around the world in no time at all? How do email and texting work, and how private are they? Why is spam so easy to send and so hard to get rid of? Do cell phones report where you are all the time? Who is tracking you online and on your phone, and why does that matter? Can your face be recognized in a crowd? Who knows that it's *your* face? Can hackers take over your car? How about self-driving cars? Can we defend our privacy and security, or should we just give up? By the end of the book, you should have a decent idea of how computer and communications systems work, how they affect you, and how you can strike a balance between using helpful services and protecting your privacy.

There are only a handful of fundamental ideas, which we will discuss in much more detail in the rest of the book.

First is the *universal digital representation of information*. Complex and sophisticated mechanical systems like those that stored documents, pictures, music and movies for much of the 20th century have been replaced by a single uniform storage mechanism. Information is represented digitally—as numeric values—rather than in specialized forms like colored dyes embedded in plastic film or magnetic patterns on vinyl tape. Paper mail gives way to digital mail. Paper maps yield to digital ones. Paper documents are replaced by online databases. All those disparate analog representations have been replaced by a common low-level representation in which everything is just numbers: digital information.

Second is the *universal digital processor*. All of this digital information can be processed by a single general-purpose device, the digital computer. Digital computers that process the uniform digital representation have replaced the elaborate and complicated mechanical devices that process analog representations. As we'll see, computers are all equally capable in what they can compute, differing only in how fast they operate and how much data they can store. A smartphone is a computer of great sophistication, with as much computing power as a laptop. Thus more and more of what might once have been limited to desktop or laptop computers has found its way onto phones, and this process of convergence is accelerating.

Third is the *universal digital network*. The Internet connects the digital computers that process the digital representation; it connects computers and phones to mail, search, social networks, shopping, banking, news, entertainment, and everything else. Most of the world's population has access to this network. You can exchange email with anyone, regardless of where they might be or how they choose to access their

mail. You can search, comparison shop, and purchase from your phone, laptop, or tablet. Social networks keep you in touch with friends and family, again from your phone or computer. You can watch endless entertainment, often for free. “Smart” devices monitor and control systems in your home; you can talk to them to ask them questions or tell them what to do. There’s a worldwide infrastructure that makes all these services work together.

Fourth, an immense amount of *digital data* is continuously being collected and analyzed. Maps, aerial photographs, and street-level views of much of the world are freely available. Search engines tirelessly scan the Internet so they can answer queries efficiently. AI agents just as tirelessly scan the Internet to scoop up any kind of information that might be useful in training models. Millions of books are available in digital form. Social networks and sharing sites maintain enormous amounts of data for and about us. Both online and brick-and-mortar stores and services provide access to their wares while quietly recording everything we do when we visit them, aided and abetted by search engines, social networks and our phones. For all of our online interactions, Internet service providers log the connections we make, and likely much more. Governments spy on us all the time, to an extent and at a level of detail that would have been impossible two or three decades ago.

All of this is changing rapidly because digital technological systems continue to get smaller, faster, cheaper and more powerful. New phones with fancier features, better screens, and more interesting applications arrive continuously. New gadgets appear all the time; the most useful find their functionality subsumed into phone apps. This is a natural by-product of digital technology, in which any technological development leads to improvement across the board for digital devices: if some change makes it possible to handle data cheaper, faster or in larger quantity, all devices will benefit. As a result, digital systems are pervasive, an integral part of our lives both visibly and behind the scenes.

This progress must surely seem like a good thing, and indeed in most ways it is. But there are ominous clouds around the silver lining. One of the most obvious and perhaps the most worrying to individuals is the impact of technology on personal privacy. When you use your phone to search for some product and then visit store web sites, all parties keep records of what you visited and what you clicked on. They know who you are because your phone identifies you uniquely. They know where you are because your phone reports its location all the time. The phone company records this information and may sell it. With GPS, the Global Positioning System, you can be located to within five to ten meters; if you enable location services on your phone, that information is available to apps, and they too can use or sell that information. In fact, it’s even worse: disabling location services only prevents apps from using GPS data; it does not prevent the phone’s operating system from collecting and uploading the data, which it could do by the cell network, Wi-Fi or Bluetooth.

You’re being watched in real life as well as online. Face recognition technology can identify you on the street or in a store. Traffic cameras scan your license plates and know where your car is; so do electronic toll-collection systems. Internet-connected smart thermostats, voice responders, door locks, baby monitors and security cameras are surveillance devices that we have invited into our homes. The tracking

that we permit today without even thinking about it makes the monitoring in George Orwell's *1984* look casual and superficial.

The records of what we do and where we do it may well live forever. Digital storage is so cheap and data is so valuable that information is rarely discarded. If you post something embarrassing online or send mail that you subsequently regret, it's too late. Information about you can be combined from multiple sources to create a detailed picture of your life, and is available to commercial, government and criminal interests without your knowledge or permission. It is likely to remain available indefinitely and could surface to harm you at any time in the future.

The universal network and its universal digital information have made us vulnerable to strangers to a degree never imagined a decade or two ago. As Bruce Schneier says in his excellent book *Data and Goliath*, "Our privacy is under assault from constant surveillance. Understanding how this occurs is critical to understanding what's at stake."

The societal mechanisms that protect our privacy and our property have not kept up with the rapid advances in technology. Thirty years ago, I dealt with my local bank and other financial institutions by physical mail and occasional personal visits. Accessing my money took time and it left an extensive paper trail; it would have been difficult for someone to steal from me. Today, I am forced to deal with financial institutions mostly through the Internet. I can access my data conveniently, but it's possible that through some blunder on my part or a screwup by one of these companies, someone on the far side of the world could clean out my account, steal my identity, ruin my credit rating, and who knows what else, in no time at all and with little chance of recourse.

This book is about understanding how digital systems work and how they are changing our lives. Of necessity it's a snapshot, so you can be certain that ten years from now, today's systems will seem clunky and antiquated. Technological change is not an isolated event but an ongoing process—rapid, continuous, and accelerating. Fortunately, the basic ideas of digital systems will remain the same, so if you understand those, you'll understand tomorrow's systems too, and you'll be in a better position to deal with the challenges and the opportunities that they present.

Part I

Hardware

“I wish to God these calculations had been executed by steam.”

Charles Babbage, 1821, quoted in Harry Wilmot Buxton,
Memoir of the Life and Labours of the Late Charles Babbage, 1872.

Hardware is the solid, visible part of computing: devices and equipment that you can see and put your hands on. The history of computing devices is interesting, but I will only mention a little of it here. Some trends are worth noting, however, especially the exponential increase in how much circuitry and how many devices can be packed into a given amount of space, for a fixed price. As digital equipment has become cheaper and more powerful, widely disparate mechanical systems have been superseded by much more uniform electronic ones.

Computing machinery has a long history, though most early computational devices were specialized, often for predicting astronomical events and positions. For example, one (unproven) theory holds that Stonehenge was an astronomical observatory. The Antikythera mechanism, from about 100 BCE, is an astronomical computer of remarkable mechanical sophistication. Arithmetic devices like the abacus have been used for millennia, especially in Asia. Seventeenth century mathematicians like Pascal and Leibniz built mechanical calculators that could do basic arithmetic. The slide rule was invented in the early 1600s, not long after John Napier’s description of logarithms. I used one as an undergraduate engineering student in the 1960s, but slide rules are now curiosities, replaced by calculators and computers, and my painfully acquired expertise is useless.

The most relevant precursor to today’s computers is the Jacquard loom, which was invented in France by Joseph Marie Jacquard around 1800. The Jacquard loom used rectangular cards with multiple rows of holes that specified weaving patterns. The Jacquard loom thus could be “programmed” to weave a wide variety of different patterns under the control of instructions that were provided on punched cards; changing the cards caused a different pattern to be woven. The creation of these labor-saving machines for weaving led to widespread social disruption as weavers were put out of work; the Luddite movement in England in 1811–1816 was a violent

protest against mechanization. Modern computing technology has similarly led to disruption; for example, we are in the early stages of major changes that will be caused by artificial intelligence.

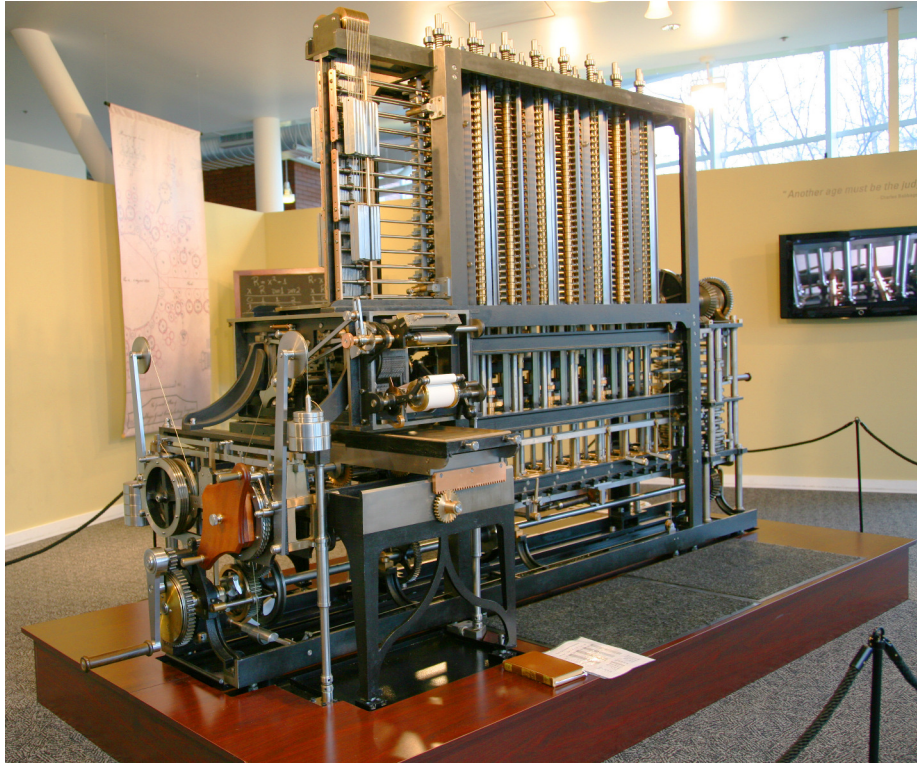


Figure I.1: Modern implementation of Babbage's Difference Engine.

Computing in today's sense began in England in the mid-19th century with the work of Charles Babbage. Babbage was a mathematician and scientist who was interested in navigation and astronomy, both of which required tables of numeric values for computing locations. Babbage spent much of his life trying to build computing devices that would mechanize the tedious and error-prone manual arithmetic calculations needed to create the tables, and even to print them. You can sense his exasperation in the quotation on the previous page. For a variety of reasons, including alienating his financial backers, he never succeeded in his ambitions, but his designs were sound. Modern implementations of some of his machines, built with tools and materials appropriate to his time, can be seen in the Science Museum in London; the version in the figure above was on display in the Computer History Museum in Mountain View, California.

Babbage encouraged a young woman, Augusta Ada Byron, the daughter of the poet George Byron, and later Countess of Lovelace, in her interests in mathematics and his computational devices. Lovelace wrote detailed explanations of the use of Babbage's Analytical Engine (the most advanced of his planned devices) for



Figure I.2: Ada Lovelace. Detail from 1836 portrait by Margaret Sarah Carpenter.

scientific computation and speculated that machines could do non-numeric computation as well, such as composing music.

Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent.

Ada Lovelace is sometimes called the world's first programmer, perhaps an overstatement, but she definitely saw more clearly than others the potential of computing devices to do more than crunch numbers. The Ada programming language is named in her honor.

Herman Hollerith, working with the US Census Bureau in the late 1800s, designed and built machines that could tabulate census information far more rapidly and reliably than could be done by hand. Using ideas from the Jacquard loom, Hollerith used holes punched in stiff paper cards to encode census data in a form that could be processed by his machines. Famously, the 1880 census had taken eight years to tabulate, but with Hollerith's punch cards and tabulating machines, the 1890 census results took only one year to prepare, instead of the predicted 10 years or more. Hollerith founded a company that in 1924 became, through mergers and acquisitions, International Business Machines, which we know today as IBM.

Babbage's machines were complex mechanical assemblies of gears, wheels, levers and rods. The development of electronics in the 20th century made it possible to imagine computers that did not rely on mechanical components. The first significant one of these all-electronic machines in the US was ENIAC, the Electronic Numerical Integrator and Computer, which was built from 1944 to 1946 at the University of Pennsylvania in Philadelphia, by Presper Eckert and John Mauchly. ENIAC occupied a large room and required 140 kilowatts of electric power, enough

to power 100 homes; it could do about 5,000 additions in a second. It was intended to be used for ballistics computations and the like, but it was not completed until well after the end of World War II. (Parts of ENIAC are on display in the Moore School of Engineering at the University of Pennsylvania.)

Babbage saw clearly that a computing device could store its operating instructions and its data in the same form, which is the insight that makes modern computing possible. But ENIAC did not store instructions in memory along with data; instead it was programmed by a slow and tedious process of setting up connections through switches and re-cabling. The first computers that truly stored programs and data together were built in England, notably EDSAC, the Electronic Delay Storage Automatic Calculator, at the University of Cambridge in 1949.

Early electronic computers used vacuum tubes as computing elements. Vacuum tubes are electronic devices roughly the size and shape of a cylindrical light bulb (see Figure 1.7 in the next chapter); they were expensive, fragile, bulky, and power hungry. With the invention of the transistor in 1947, and then of integrated circuits in 1958, the modern era of computing really began. These technologies have allowed electronic systems to steadily become smaller, cheaper and faster.

The next three chapters describe computer hardware, focusing more on the logical architecture of computing systems than on the physical details of how they are built. The basic architecture has been largely unchanged for decades, while the hardware implementation has changed to an astonishing degree. The first chapter is an overview of the structure and components of a computer. The second chapter shows how computers represent information with bits, bytes and binary numbers. The third chapter explains how computers actually compute: how they process the bits and bytes to make things happen.

1

What Is a Computer?

“Inasmuch as the completed device will be a general-purpose computing machine it should contain certain main organs relating to arithmetic, memory-storage, control and connection with the human operator.”

Arthur W. Burks, Herman H. Goldstine, John von Neumann, “Preliminary discussion of the logical design of an electronic computing instrument,” 1946.

Let’s begin our discussion of hardware with an overview of what a computer is. We can look at a computer from at least two viewpoints: the logical or functional organization—what the pieces are, what they do and how they are connected—and the physical structure—what the pieces look like and how they are built. The goal of this chapter is to understand what a computer is, see what’s inside, learn what each part does, and get a sense of what the myriad acronyms and numbers mean.

Think about your own computing devices. Many readers will have some kind of “PC,” that is, a laptop or desktop computer descended from the Personal Computer that IBM first sold in 1981, running some version of the Windows operating system from Microsoft. Others will have an Apple Macintosh that runs a version of the macOS operating system. Still others might have a Chromebook running ChromeOS that relies on the Internet for most of its storage and computation. More specialized devices like smartphones, tablets and ebook readers are also powerful computers. These all look different and when you use them they feel different as well, but underneath the skin, they are fundamentally the same. We’ll talk about why.

There’s a loose analogy to cars. Functionally, cars have been the same for well over a hundred years. A car has an engine that uses some kind of fuel or energy source to make the engine run and the car move. It has a steering wheel that the driver uses to control the car. There are places to store the fuel and places to store the passengers and their goods. Physically, however, cars have changed greatly over a century: they are made of different materials, and they are faster, safer, and much more reliable and comfortable. Their propulsion systems may be electric motors instead of internal combustion engines. There’s a world of difference between my first car, a well-used 1959 Volkswagen Beetle, and a Ferrari, but either one will carry

me and my groceries home from the store or across the country, and in that sense they are functionally the same. (For the record, I have never even sat in a Ferrari, let alone owned one, so I'm speculating about whether there's room for the groceries. I did park next to one once—Figure 1.1.)



Figure 1.1: The closest I've ever come to a Ferrari.

The same is true of computers. Logically, today's computers are very similar to those of the 1950s, but the physical differences go far beyond the kinds of changes that have occurred with the automobile. Today's computers are much smaller, cheaper, faster and more reliable than those of 75 years ago, easily a million or even a billion times better in some properties. Such improvements are the fundamental reason why computers are so pervasive.

The distinction between the functional behavior of something and its physical properties—the difference between what it does and how it's built or works inside—is important. For computers, the “how it's built” part changes at an amazing rate, as does how fast it runs, but the “what it does” part is quite stable. This distinction between an abstract description and a concrete implementation will come up repeatedly in what follows.

I sometimes do a survey in my class in the first lecture. How many have a PC? How many have a Mac? The ratio was fairly constant at 10 to 1 in favor of PCs in the early 2000s, but changed rapidly over a few years, and for the past twenty years Macs have accounted for well over three quarters of the computers. This is not typical of the rest of world, however, where PCs are likely to dominate.

Is the ratio unbalanced because one is superior to the other? If so, what changed so dramatically in such a short time? I ask my students which kind is better, and for objective criteria on which to base that opinion. What led you to your choice when you bought your computer?

Naturally, price is one answer. PCs tend to be cheaper, the result of fierce competition in a marketplace with many suppliers. A wider range of hardware add-ons, more software, and more expertise are all readily available. This is an example of what economists call a *network effect*: the more other people use something, the more useful it will be for you, roughly in proportion to how many others there are.

On the Mac side are perceived reliability, quality, esthetics, and a sense that “things just work,” for which many consumers are willing to pay a premium.

The debate goes on, with neither side convincing the other, but it raises some good questions and helps to get people thinking about what is different between various kinds of computing devices and what is really the same.

There’s an analogous debate about phones. Almost everyone has a “smartphone” that can run programs (“apps”) downloaded from Apple’s App Store or the Google Play Store. The phone serves as a browser, a mail system, a watch, a camera, a music and video player, a voice recorder, a map, a navigator, a comparison shopping tool, and even occasionally as a device for conversation. Typically more than three quarters of my students have iPhones; the rest have Android phones from one of many suppliers. iPhones are more expensive but offer smooth integration with Apple’s ecosystem of computers, tablets, watches, music players, accessories, and cloud services, another example of a network effect. Rarely, someone admits to having only a “feature phone,” which is defined as a phone that has no features beyond the ability to make phone calls. My sample is for the US and a comparatively affluent environment; in other environments and other parts of the world, Android phones would be much more common.

Again, people have good reasons—functional, economic, social, esthetic—for choosing one kind of phone over others but underneath, just as for PCs versus Macs, the hardware that does the computing is very similar. Let’s look at why.

1.1 Logical Construction

If we were to draw an abstract picture of a simple generic computer—its logical or functional architecture—it would look like the diagram in Figure 1.2 for both a Mac and a PC: a processor, some primary memory, some secondary storage, and a variety of other components, all connected by a set of wires called a *bus* that carries information between them.

If instead we drew this picture for a phone or tablet, it would be similar, though mouse, keyboard and display are combined into one component, the screen, and there are hidden components like a compass, an accelerometer, and a GPS receiver for determining your physical location.

The basic organization—a processor, memory and storage for instructions and data, and input and output devices—has been standard since the 1940s. It’s called the *von Neumann architecture*, after John von Neumann, who described it in the 1946 paper quoted above. Though there is still occasional debate over whether von

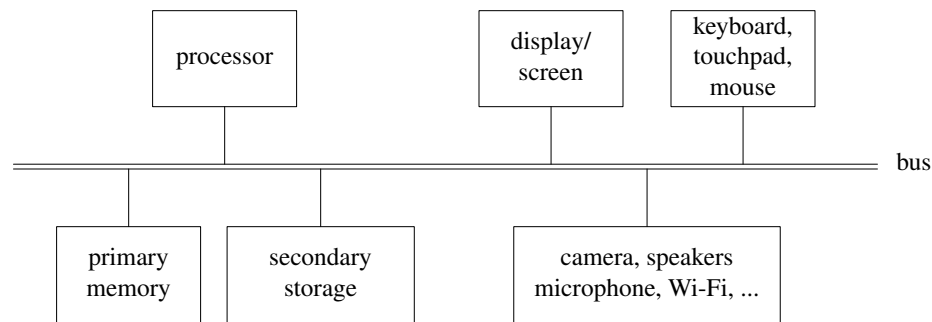


Figure 1.2: Architectural diagram of a simple idealized computer.

Neumann gets too much credit for ideas that originated with others, the paper is so clear and insightful that it is well worth reading even today. For example, the quotation at the beginning of this chapter is the first sentence of the paper. Translated into today’s terminology, the processor provides arithmetic and control, the primary memory and secondary storage are memory-storage, and the keyboard, mouse and display interact with the human operator.

A note on terminology: the processor has historically been called the *CPU* or *central processing unit*, but is now more often simply “processor.” The primary memory is called *RAM* or *random access memory*, and secondary storage is *disk* or *drive*, reflecting different physical implementations. I’ll mostly use the words processor, memory and storage, but with occasional lapses into the older terms.

1.1.1 Processor

The processor is the brain, if a computer could be said to have such a thing. The processor does arithmetic, moves data around, and controls the operation of the other components. The processor has a limited repertoire of basic operations that it can perform but it does them blazingly fast, billions per second. It can decide what operations to do next based on the results of previous computations, so it is to a considerable degree independent of its human users. We will spend more time on this component in Chapter 3 because it’s so important.

If you go to a store or shop online to buy a computer, you’ll find most of these components mentioned, usually accompanied by mysterious acronyms and equally mysterious numbers. For example, you might see a processor described as a “2.2 GHz dual-core Intel Core i7,” as it is for one of my older computers. What’s that? Intel makes the processor and “Core i7” is Intel’s trademarked name for an extensive line of processors. This particular processor has two processing units in a single package; in this context, lower case “core” has become a synonym for “processor.” A core is a processor in its own right, but the CPU may have several cores that can work together or independently to compute faster. For most purposes, however, it’s sufficient to think of the combination as “the processor,” no matter how many cores it might have.

“2.2 GHz” is the more interesting part. Processor speed is measured, at least approximately, in terms of the number of operations or instructions or parts thereof that it can do in a second. The processor uses an internal clock, rather like a heartbeat or the ticking of a clock, to step through its basic operations. One measure of speed is the number of such ticks per second. One beat or tick per second is called one *hertz* (abbreviated *Hz*), after the German engineer Heinrich Hertz, whose 1888 discovery of how to produce electromagnetic radiation led directly to radio and other wireless systems. Computers of the 1970s had clock speeds measured in millions of ticks per second, or megahertz, or MHz. Computers today typically run at billions of hertz, or gigahertz, or GHz; my quite ordinary 2.2 GHz processor is zipping along at 2,200,000,000 ticks per second. The human heartbeat is around 1 Hz or almost 100,000 beats per day, which is around 30 million per year. So each core in my processor does in 1 second the number of beats my heart would do in 70 years.

This is our first encounter with some of the numerical prefixes like mega and giga that are so common in computing. “Mega” is one million, or 10^6 ; “giga” is one billion, or 10^9 , and pronounced with a hard “g” as in “gig.” We’ll see more units soon enough, and there is a complete table at the beginning of the glossary.

1.1.2 Primary memory

The primary memory stores information that is in active use by the processor and other parts of the computer; its contents can be changed by the processor. The primary memory stores not only the data that the processor is currently working on, but also the instructions that tell the processor what to do with that data. This is a crucially important point: by loading different instructions into memory, we can make the processor do a different computation. This makes the stored-program computer a general-purpose device. The same computer can run a word processor and a spreadsheet, surf the web, send and receive email, keep up with friends on Instagram, do your taxes, and play music, all by placing suitable instructions in the memory. The importance of the stored-program idea cannot be overstated.

The primary memory provides a place to store information while the computer is running. It stores the instructions of programs that are currently active, like Word, Photoshop or a browser. It stores their data—the documents being edited, the pictures on the screen, the music that’s currently playing. It also stores the instructions of the operating system—Windows, macOS or something else—that operates behind the scenes to let you run multiple applications at the same time. We’ll talk about applications and operating systems in Chapter 6.

The primary memory is called *random access* because the processor can access the information stored at any place within it as quickly as in any other; to oversimplify a little, there’s no speed penalty for accessing memory locations in any particular order. Though they passed from the scene long ago, you might remember VCR tapes, where to watch the end of a movie, you had to fast-forward (slowly!) over everything from the beginning; that’s called *sequential access*.

Most RAM is *volatile*, that is, its contents disappear if the power is turned off, and all of the currently active information is lost. That’s why it’s prudent to save your work often, especially on a desktop machine, where tripping over the power cord

(continued...)

Index

“Any inaccuracies in this index may be explained by the fact that it has been sorted with the help of a computer.”

Donald E. Knuth, *The Art of Computer Programming*,
Volume 3, *Searching and Sorting*, 1973.

- 1-click patent, Amazon 91
- 1Password 205
- 3-2-1 backup strategy 206
- 3Com 150
- 3G cell phone 155
- 4G, 5G cell phone 157, 282
- 64-bit memory address 40
- 802.11 wireless standard 152, 157, 162, 282
- abstraction 12, 18, 101, 106, 113, 117, 140, 268
- accumulator 44, 51, 53
- acknowledgment, TCP 176
- active content 193
- Activity Monitor, macOS 101
- Ad Exchange, Google 218
- Ada Lovelace 8–9
- Ada programming language 9
- address translation (NAT), network 169
- address, Ethernet 150
 - IP 165, 168, 285
 - IPv4 165
 - IPv6 165, 169
- Adleman, Leonard 254
- Advanced Audio Coding (AAC) 33, 160, 282
- advertising, Google 271
- AES encryption standard 252, 282
- Afghanistan Internet shutdown 208
- age verification 271
- AI winter 235
- Aiken, Howard 80
- airplanes, computers in 60
- Akamai 175
- alert function 123, 129
- Alexa 233
- algorithm complexity 71, 139, 283
- algorithm complexity graph 69, 133
- algorithm vs program 75
- algorithm, ATM checksum 161
 - binary search 66, 282
 - definition of 64, 139, 282
 - divide and conquer 66, 68, 169
 - exponential 71, 254, 284
 - ISBN checksum 161
 - linear 65
 - MD5 message digest 256, 285
 - $N \log N$ 69
 - PageRank 214
 - quadratic 68–69, 286
 - Quicksort 68, 91
 - recipe as 63
 - rot13 encryption 250
 - RSA public-key 254, 286
 - selection sort 67
 - SHA-3 message digest 256, 286
 - tax form as 63
- Allen, Paul 80
- allocation, spectrum 152, 162
- AlphaGo, AlphaZero 240
- Altair computer 80
- AltaVista search engine 212
- Alto computer 149
- Amazon xiii, 122, 167, 177, 192, 255, 271
 - Amazon 1-click patent 91
 - Amazon Echo 221
 - Amazon Fire TV 221
 - Amazon Kindle 93, 101, 108
 - Amazon Ring 246, 270
 - Amazon transparency report 230
 - Amazon Web Services (AWS) 104
 - AMD processor 52
 - amplitude modulation (AM) 151, 282
 - AMSL 23
 - analog hole 90
 - analog vs digital 27
 - analog watch 28
 - analog-digital conversion 29
 - Android operating system 96
 - Android phone 13, 94, 101, 107
 - Android SDK 86
 - Android Studio IDE 86
 - Anguilla, .ai 168
 - Anna Kournikova virus 195–196
 - anonymizing data 226
 - Anthropic 90, 183, 234
 - anti-virus software 206
 - Antikythera mechanism 7
 - AOL query logs 226
 - Apache web server 96
 - app 13, 99, 115, 282
 - Apple 11, 86, 91, 99, 101, 117
 - Apple AirTag 152
 - Apple iCloud 264
 - Apple Neo 52
 - Apple Time Machine 206
 - Apple TV 221
 - Application Programming Interface (API) 86, 93, 122, 128, 140, 223, 282
 - application, date 115
 - application-level protocol 174
 - architecture, instruction set 51
 - architecture, von Neumann 13, 24
 - arithmetic, binary 38
 - ARM processor 52, 77, 117, 282
 - ARPANET 163–164
 - ASCII characters, table of 34
 - ASCII standard 34, 40, 57, 95, 158, 178, 282
 - ASCII with parity, table of 161
 - assembler 76, 139, 282
 - assembly programming language 76
 - assignment statement 125–126
 - asynchronous event 116
 - AT&T 144–145, 153, 231
 - ATM checksum algorithm 161
 - attack, brute force 204, 252
 - dictionary 195, 252
 - escalation of privilege 200
 - man-in-the-middle 143, 155, 203, 250, 253, 285

- phishing xii, 198, 286
- SIM-swapping 201
- social engineering 197, 199, 287
- spear phishing 199, 286
- SQL injection 202
- supply-chain 61
- zero-day 88
- attention mechanism 243
- Austen, Jane 136
- authentication, key 255
- password 178, 185, 191, 200, 203–204, 265
- two-factor 205
- Authors Guild vs Google 182
- autonomous system 171
- Autopilot, Tesla 60, 93
- autorun 196
- Babbage, Charles 7–8
- backdoor, cryptographic xiii, 252, 264, 282
- backup strategy, 3-2-1 206
- Backus, John 79
- bandwidth 147–148, 157, 282
- barcode 18, 161
- Bardeen, John 20
- base 2 logarithm 66
- base station, cell phone 153
- base station, wireless 151, 282
- BASIC programming language 80
- Bell Labs 20, 80, 100, 144
- Bell, Alexander Graham 143, 147
- Bengio, Yoshua 239
- Berners-Lee, Tim 188
- best effort 175
- Betamax vs VHS 94
- Bezos, Jeff 91, 207
- big data 225
- bigit 35
- binary addition table 39
- binary arithmetic 38
- binary digit 35
- binary multiplication table 39
- binary numbers 37, 57, 282
- binary search algorithm 66, 282
- binary to decimal conversion 38
- Bing search engine 74, 189
- bit rot 88, 282
- bit*, origin of 27, 35
- Bitcoin blockchain 262, 282
- Bitcoin protocol 181, 201, 261, 282
- bits, meaning of 35, 37, 282
- BitTorrent protocol 181, 282
- Blacklight* tracker monitor 221
- block, disk 110
- Blu-ray vs HD-DVD 94
- blue screen of death 101
- Bluetooth 108, 152, 282
- Boeing 737 MAX 60
- Boeing MCAS 60
- Boggs, David 149
- booting 105
- bot, botnet 200, 282
- branch instruction 47
- branch instruction, conditional 47
- Branwen, Gwern 246
- Brattain, Walter 20
- breach, Capital Health 202
- Equifax 202
- RSA 205
- SecurID 205
- TransUnion 202
- UnitedHealthcare 61, 202
- VPN 204
- bricking 183, 282
- Brin, Sergey 212
- broadcast network 147, 186
- Brooks, Frederick P. 99–100
- browser fingerprinting 220, 258
- browser, Chrome 81, 116, 123, 188
- Chromium 96
- DuckDuckGo 206
- Edge 116, 188
- Firefox 81, 96, 116, 123, 188, 206, 259
- Internet Explorer 118, 188
- Mosaic 188
- Netscape Navigator 118, 188, 193
- Safari 116, 123, 188, 204
- Tor 207, 259
- browserleaks.com 220
- brute force attack 204, 252
- Buchholz, Werner 39
- bug 86, 128–129, 140, 195, 283
- bug picture 86
- Burks, Arthur 11, 43, 53
- bus 13, 18, 283
- Butler, Samuel 1
- byte 16, 39, 283
- bzip2 compression 158
- C programming language 80–81, 83, 95, 116, 193
- C# programming language 96
- C++ programming language 80–81, 83, 95, 116
- cable modem 148, 151, 283
- cache 51, 53, 169, 283
- Caesar cipher 250
- California Consumer Privacy Act (CCPA) 222
- California Delete Request and Opt-out Platform (DROP) 222
- canvas fingerprinting 220
- Capital Health breach 202
- CAPTCHA 55, 283
- cars, computers in 59
- Cascading Style Sheets (CSS) 191
- CD encoding 32
- CD-ROM 32, 113
- cell phone base station 153
- cell phone cells diagram 154
- cell phone frequency 154
- cell phone, 3G 155
- cell phone, 4G, 5G 157, 282
- central processing unit (CPU) 14, 43, 283, 286
- Cerf, Vinton 177
- CERN 188
- certificate 257, 283
- certificate authority 257
- Chappe, Claude 142
- chatbot, Eliza 241
- ChatGPT xiii, xv–xvi, 1, 90, 234, 241
- checksum algorithm, ATM 161
- checksum algorithm, ISBN 161
- chess 233, 240
- Children’s Online Privacy Protection Act (COPPA) 184
- chip 21, 283
- Chrome browser 81, 116, 123, 188
- Chromebook 11, 116
- Chromecast 221
- ChromeOS operating system 11, 116, 283
- Chromium browser 96
- circuit board 19
- Claude xvi, 234, 242
- Clay Mathematics Institute 73
- cleanroom development 90, 103
- Clearview AI 240
- client-server 181, 187, 191, 283, 286
- cloud computing xiii, 25, 104, 114, 227, 283
- cloud picture 227
- clustering, *k*-means 237
- coaxial cable 151
- COBOL programming language 79
- Cocks, Clifford 254
- code size, Google 85
- Codecademy 122
- Cohen, Bram 181
- Coinbase 263
- Colab, Google 122, 130, 136
- color representation, RGB 30, 40, 286
- Comcast 145
- command, nslookup 169
- telnet 178
- traceroute 172
- whois 168
- comments 46
- Communications Decency Act 145
- compact disc 32
- compilation process diagram 78
- compiler 77, 129, 136, 139, 283
- complexity graph, algorithm 69, 133
- complexity, algorithm 71, 139, 283
- components, software 85
- compression, bzip2 158
- data 155, 158, 229, 283
- GIF 158, 191, 284
- Huffman code 158
- JPEG 33, 74, 159, 191, 283, 285
- lossless 158
- lossy 159
- MP3 17, 33, 74, 160, 180, 283, 285
- MPEG 33, 74, 159, 285
- PNG 74, 158, 286
- ZIP 74, 158
- computer architecture 51
- computer architecture diagram 13
- Computer Fraud and Abuse Act 195
- computer vision 239
- computer, Altair 80
- Alto 149
- EDSAC 10, 76
- ENIAC 9, 20
- IBM 7094 xi
- IBM PC 11, 23
- stored-program 15, 55
- Toy 44, 76
- computers in airplanes 60
- computers in cars 59
- computers in medical imaging 61
- computers in voting machines 61
- concatenation, text 125, 127
- conditional branch instruction 47
- Confucius 121
- conversion, analog-digital 29
- binary to decimal 38
- decimal to binary 38
- Cook, Stephen 72
- Cook, Tim 264
- cookie 192, 217, 283
- cookie, third-party 197, 206
- cookies, tracking 192, 217
- cooking robot 75
- copyright 89, 95, 181
- copyright lifetime 90, 270
- Count of Monte Cristo* 143
- CoverYourTracks 221, 260
- Covid-19 61, 235

- crawl, crawler 212
- Creative Commons license 96
- cryptocurrency 262, 283
- cryptographic backdoor xiii, 252, 264, 282
- cryptographic hash 256
- cryptographic key 251
- cryptography 71, 74, 250
- ctof function 87
- CTSS operating system 100
- cyber attack, Ukraine 61
- cyber warfare 61, 196, 270
- dark web 283
- data compression 155, 158, 229, 283
- Data General Corporation 100
- data mining 209
- data science 209
- data structure 64, 82, 86
- date application 115
- DDoS *see* denial of service attack
- debugging 86, 88, 128
- decimal to binary conversion 38
- declaration, variable 124, 126, 283, 287
- deep learning 234, 283
- deep packet inspection 221
- definition of algorithm 64, 139, 282
- deidentification 226
- delay 147
- denial of service attack (DDoS) 203
- Dept of Justice vs Microsoft 118, 188
- DES encryption standard 252, 283
- device driver 107, 283
- DHCP protocol 166
- diagram, cell phone cells 154
 - compilation process 78
 - computer architecture 13
 - Ethernet packet 150
 - file system organization 110
 - IPv4 packet 175
 - operating system and libraries 107
 - protocol layers 174
 - search engine 212
 - sound wave 31
 - TCP segment 176
 - TCP/IP connection 177
 - Toy program in memory 49
 - virtual operating system 104
- dictionary attack 195, 252
- Diffie, Whitfield 253
- Digital Equipment Corporation (DEC) 100
- Digital Millennium Copyright Act (DMCA) 182–183, 283
- digital representation 32
- digital rights management (DRM) 90
- digital signature 255
- digital television 162
- digital vs analog 27
- digital watch 28
- directory *see* folder
- disc, compact 32
- disk block 110
- disk, floppy 23
- distributed computing 54
- divide and conquer algorithm 66, 68, 169
- division by zero 65, 128
- DMCA safe harbor 182
- Do Not Track 218
- DOGE xii, 264, 270
- domain name 166, 189, 283
- domain name registrar 167, 170, 286
- Domain Name System (DNS) 166–167, 174, 283
- domain name, internationalized 168
- domain name, Unicode 168
- domain, top-level 167
- DoS *see* denial of service attack
- dotted decimal notation 165
- DoubleClick 218
- doxx, doxxing 283
- Dropbox 114, 227
- DSL 284
- DuckDuckGo browser 206
- DuckDuckGo search engine 215
- Dumas, Alexandre 143
- dumb network 185
- DVD 32, 40, 113
- Dynamic Host Configuration Protocol (DHCP) 166
- eavesdropping 186, 254
- echo chamber 207
- Eckert, J. Presper 9
- Edge browser 116, 188
- Edison, Thomas 30, 86
- EDSAC computer 10, 76
- Eich, Brendan 82, 193
- electrical telegraph 143
- Electronic Frontier Foundation (EFF) 221
- Eliza chatbot 241
- Ellis, James 254
- email privacy 230
- embedded system 59, 81
- embedding 242
- encoding, CD 32
 - Punycode 170
 - RBG 40
 - URL 189
 - UTF-8 58, 287
- encryption 90, 148, 150, 207
- encryption (WEP, WPA), wireless 162
- encryption algorithm, `rot13` 250
- encryption standard, AES 252, 282
- encryption standard, DES 252, 283
- encryption, end-to-end 204, 264
- encryption, iPhone 264
- end-to-end encryption 204, 264
- end-to-end principle 185
- English wine measures 41
- ENIAC computer 9, 20
- Enigma 55, 251
- Equifax breach 202
- error correction 74, 114, 155, 160
- error detection 160
- error, semantic 79
- error, syntax 78
- escalation of privilege attack 200
- Ethernet 149, 284
- Ethernet address 150
- Ethernet packet 165
- Ethernet packet diagram 150
- Ethernet range 151
- EULA 92, 284
- European Union xiii, 222, 229, 271
- exa 54, 209
- exchange point (IXP), Internet 171, 285
- expert system 235
- exponential algorithm 71, 254, 284
- extension, Ghostery 206, 217, 231
- NoScript 194, 207
- Privacy Badger 206
- uBlock Origin 206, 231
- F# programming language 96
- face recognition 3, 234, 239
- Facebook xii, 122, 145, 167, 211, 222–223, 271
- Facebook Like 224
- Facebook transparency report 230
- factoring 254
- factoring challenge, RSA 254
- Fanning, Shawn 180
- feature phone 13
- features, machine learning 239
- Federal Communications Commission (FCC) 145, 152
- Federal Election Commission (FEC) 225
- Federal Trade Commission (FTC) 183, 204
- fetch-decode-execute cycle 44, 50
- Feynman Algorithm 63
- fiber, optical 22, 144, 147, 149, 157, 163, 284
- File Recovery program, Windows 112
- file sharing 180
- file system 17, 108, 284
- file system organization diagram 110
- file system, network 114
 - RAID 114
 - SD card 113
 - USB 113
- filter bubble 207, 284
- fingerprinting, browser 220, 258
- fingerprinting, canvas 220
- Firefox browser 81, 96, 116, 123, 188, 206, 259
- firewall 207–208, 272, 284
- firkin 41
- first sale doctrine 93
- flash memory 16, 113, 284
- flicker 33
- floating-point operation (flop) 54
- Flock surveillance 270
- floppy disk 23
- floppy disk virus 196
- folder 109, 284
- folder contents 110
- folder, trash 112
- for loop, Python 133
- form tag 191
- form, HTML 191
- Fortran programming language 79
- forward secrecy 256
- Fox News 2
- Frankenpine 155
- Free Software Foundation 95
- frequency modulation (FM) 151, 284
- frequency, cell phone 154
- ftoc function 87
- function 85
- function call 85, 284
- function, alert 123, 129
 - ctof 87
 - ftoc 87
 - parseInt 127
 - prompt 124–125, 129
- Gates, Bill 80
- gateway 164–166, 171–172, 175, 177, 185, 221, 284
- gateway vs router 164
- Gell-Mann, Murray 63
- Gemini xvi, 136, 234
- General Data Protection Regulation (GDPR) xiii, 222, 284
- Generative Pre-trained Transformer (GPT) 243, 284
- geo-tagging 216
- geolocation 223
- Ghostery extension 206, 217, 231
- GIF compression 158, 191, 284

- giga 15, 281
- GitHub 136
- Global Positioning System (GPS) 5, 153, 216, 284
- Gmail 180, 227
- GNU General Public License (GPL) 95–96, 284
- GNU project 95
- Go (game) 233, 240
- Go programming language 96
- Goldstine, Herman 11, 43, 53
- Google xiii, 96, 116, 122, 153, 167, 212, 234
- Google Ad Exchange 218
- Google Ads 214
- Google advertising 271
- Google Cloud Platform 104
- Google code size 85
- Google Colab 122, 130, 136
- Google Docs 114, 122, 228
- Google Drive 114
- Google Maps 128, 140
- Google Meet 144
- Google search engine 74
- Google Sheets 114
- Google Street View 239
- Google tag manager 219
- Google Takeout 206
- Google transparency report 230
- Gosling, James 81
- GOTO instruction 47, 126
- Government Communications Headquarters (GCHQ) 254
- graph, algorithm complexity 69, 133
- Graphics Processing Unit (GPU) 23, 52, 253, 284
- grayscale 40
- Great Firewall of China 208, 222
- GSM 284
- Gutenberg Project 136
- Hacker News 2
- hallucination 244
- hard disk, hard drive 16, 284
- Harvard Mark I, II 80, 86
- hash, cryptographic 256
- haveibeenpwned.com 205
- HD-DVD vs Blu-ray 94
- Hellman, Martin 253
- Herodotus 142
- hertz (Hz) 15, 151
- Hertz, Heinrich 15
- hexadecimal 39, 150, 165, 189, 284
- high-level programming language 77
- Hinton, Geoffrey 239
- Hoare, Tony 68
- Hollerith, Herman 9
- Hopper, Grace 79, 86, 157
- host 170
- hotspot, mobile 156
- HP printers 183
- HTML 122–123, 187, 189, 284
- HTML form 191
- HTML tag 189
- HTML5 220
- HTTP protocol 177, 187, 284
- HTTPS protocol 177, 189, 203, 206, 222, 255, 284
- Huffman code compression 158
- hypertext 188
- IBM 9, 39, 79, 96, 100, 117, 161, 167, 252
- IBM 7094 computer xi
- IBM OS/360 operating system 100
- IBM PC computer 11, 23
- IBM System/360 100
- ICANN 167, 170, 284
- iCloud, Apple 264
- IEEE 152
- if-else statement 127
- ifconfig, ipconfig 150
- IFZERO instruction 47, 126–127
- image recognition 239
- IMAP protocol 180
- iMessage 204
- img tag 190
- Immigration and Customs Enforcement (ICE) 3
- import, Python 133
- in-store beacon 223
- incognito mode 219
- infrared 151
- Instagram xii, 211, 223–224, 271
- instruction repertoire 44, 51
- instruction set architecture 51
- instruction, branch 47
- conditional branch 47
- GOTO 47, 126
- IFZERO 47, 126–127
- jump 47
- instructions, numeric values of 49
- integrated circuit (IC) 10, 21, 284
- integrated circuit wafer 21
- Intel 22
- Intel processor 14, 51, 76, 117
- intellectual property 89, 284
- interface 140, 285
- International Mobile Equipment Identity (IMEI) 154
- International System of Units (SI) 209, 281
- International Telecommunication Union (ITU) 152, 164, 211
- internationalized domain name 168
- Internet Engineering Task Force (IETF) 167
- Internet exchange point (IXP) 171, 285
- Internet Explorer browser 118, 188
- Internet of Things (IoT) xiii, 54, 108, 183–184, 207, 269
- Internet Protocol (IP) 174, 285
- Internet Service Provider (ISP) 157, 165, 171, 182, 221, 230, 258, 285
- Internet shutdown, Afghanistan 208
- Internet time 180
- Internet voting 61
- interpreter 140, 285
- interpreter, Visual Basic 196
- iOS operating system 101
- IP address 165, 168, 285
- IP packet 165
- iPad 18
- iPhone 13, 99, 101
- iPhone encryption 264
- iPhone SDK 86
- IPv4 address 165
- IPv4 packet diagram 175
- IPv4 protocol 165
- IPv6 address 165, 169
- IPv6 protocol 165
- ISBN checksum algorithm 161
- iTunes Store 91, 99
- Jacquard loom 7
- Java programming language 81, 121, 193
- JavaScript in PDF 195
- JavaScript programming language 82, 121, 193, 228, 285
- JavaScript tracking 219
- jitter 147
- John Deere 183
- JPEG compression 33, 74, 159, 191, 283, 285
- jump instruction 47
- Jupyter notebook 130
- jurisdiction 208, 271
- k*-means clustering 237
- Kahn, Robert 177
- kernel 117, 285
- Keurig 183
- key authentication 255
- key distribution 253–254
- key logger 200, 285
- key, cryptographic 251
- Khan Academy 122
- Kilby, Jack 21
- kilderkin 41
- kilo 23, 37, 281
- Kindle, Amazon 93, 101, 108
- King Charles III 178
- Knuth, Donald 41, 289
- Large Language Model (LLM) xiii, xv–xvi, 90, 136, 241
- laser 22, 32, 91
- LastPass 205
- latency 147
- layers, software 174
- LeCun, Yann 239
- Leibniz, Gottfried Wilhelm 37, 39, 41
- Let's Encrypt 258
- library, program 86, 123, 128, 140, 285
- LibreOffice 228
- license 92
- license, Creative Commons 96
- license, virtual machine 105
- LiDAR 60
- linear algorithm 65
- link rot 285
- LinkedIn xii, 211, 224, 227
- Linux operating system 96, 103, 117, 285
- list, Python 133
- location privacy 223
- LockBit 202
- logarithm 7, 66
- logarithm, base 2 66
- logic gate 20
- long-playing record (LP) 30, 32
- loop 47, 50, 53, 175, 285
- loop, *while* 126
- lossless compression 158
- lossy compression 159
- Lovelace, Ada 8–9
- Luhn, Peter 161
- machine learning 197, 234, 236
- machine learning features 239
- macOS Activity Monitor 101
- macOS operating system 11, 92, 101
- mailinator.com 207
- malware 196, 198, 285
- man-in-the-middle attack 143, 155, 203, 250, 253, 285
- Mars Climate Orbiter 84
- Mary, Queen of Scots 250
- matplotlib, Python 133
- Mauchly, John 9
- MCAS, Boeing 60
- McClure, Robert 59
- McKenzie, Archie 1
- MD5 message digest algorithm 256, 285
- meaning of bits 35, 37, 282

- medical imaging, computers in 61
- mega 15, 281
- megapixel 30
- memory (RAM), random access 15, 43, 286
- memory address, 64-bit 40
- memory, flash 16, 113, 284
- memory, virtual 101, 140, 287
- Menabrea, Luigi 9
- Merkle, Ralph 253
- message digest algorithm, MD5 256, 285
- message digest algorithm, SHA-1, SHA-2, SHA-3 256, 286
- Messenger 204
- metadata 3, 180, 206, 222, 224, 264
- Metcalfe, Robert 149–150
- micro 23, 281
- microchip 21, 285
- micron 23
- Microsoft 11, 80, 100, 105, 188, 228
- Microsoft antitrust case 118, 188
- Microsoft Azure 104
- Microsoft OneDrive 114
- Microsoft security laws 194
- Microsoft Teams 144
- milli 23, 281
- MIME 178
- Minsky, Marvin 235
- mobile hotspot 156
- Mobile Virtual Network Operator (MVNO) 155
- modem 148, 157, 285
- modem, cable 148, 151, 283
- modulation 148–149, 151
- Moldova, .md 168
- Moore's Law 22, 24, 57
- Moore, Gordon 22
- Morris worm 195
- Morris, Robert T. 195
- Morse code 158
- Morse, Samuel 143
- Mosaic browser 188
- MP3 compression 17, 33, 74, 160, 180, 283, 285
- MPAA 181
- MPEG compression 33, 74, 159, 285
- MS-DOS operating system 113
- Multics operating system 100
- Musk, Elon xii, 184, 264, 270
- Muybridge, Eadweard 33
- $N \log N$ algorithm 69
- Nakamoto, Satoshi 261
- name server 170
- name server, root 169
- nano 23, 51, 281
- Napier, John 7
- Napster 180
- National Institute of Standards and Technology (NIST) 236, 252, 256
- National Security Agency (NSA) xii, 3, 88, 165, 197, 222, 224, 252, 256, 260, 264
- Near field communication (NFC) 153, 285
- nearest-neighbor tour 72
- negative acknowledgment 176
- Nelson, Ted 188
- Nest thermostat 101
- net neutrality 145, 285
- Netflix 17, 33, 104, 145, 181, 218
- Netscape 82, 192
- Netscape Navigator browser 118, 188, 193
- network address translation (NAT) 169
- network effect 13, 271
- network file system 114
- neural network 234, 285
- Neural Processing Unit (NPU) 52
- New York City taxi dataset 226
- New York Times* 90, 153, 216, 219, 224
- NGINX web server 96
- Nietzsche, Friedrich 211
- Nigerian scam letter 198
- Nobel Prize 20–21, 56, 63, 239
- NoScript extension 194, 207
- Notepad 124
- Noyce, Robert 21
- NP problem 71
- `nslookup` command 169
- numbers, binary 37, 57, 282
- numeric values of instructions 49
- Nvidia 23
- object code 95, 193, 285
- Objective-C programming language 81
- onion router 259
- open relay 186
- open source 95, 285
- OpenAI 1, 90, 234
- OpenPGP 261
- operating system 99
- operating system and libraries diagram 107
- operating system, Android 96
- ChromeOS 11, 116, 283
- CTSS 100
- IBM OS/360 100
- iOS 101
- Linux 96, 103, 117, 285
- macOS 11, 92, 101
- MS-DOS 113
- Multics 100
- TAILS 261
- Unix 100–101, 103, 115, 117, 287
- virtual 104
- Windows 11, 40, 100, 103, 105, 107, 117, 195–196, 198, 228
- optical fiber 22, 144, 147, 149, 157, 163, 284
- optical telegraph 142
- Oracle vs Google 94
- origin of *bit* 27, 35
- Orwell, George, *1984* 6, 93
- Outlook 180
- Outlook.com 227
- $P=NP$ problem 73
- packet 150, 165, 184, 285
- packet, Ethernet 165
- packet, IP 165
- Page, Larry 212
- PageRank algorithm 214
- Panoptick 221, 258
- Parallels virtual operating system 104
- PARC, Xerox 149
- parity code 161
- `parseInt` function 127
- password authentication 178, 185, 191, 200, 203–204, 265
- password manager 205
- patent 91, 164, 271
- patent troll 91
- patent, Amazon 1-click 91
- Patriot Act 224, 229
- PC *see* IBM PC computer
- PDF, JavaScript in 195
- peer-to-peer 181, 262, 286
- personally identifiable information 4, 202, 216, 222, 224, 226
- peta 36, 54, 281
- phishing attack xii, 198, 286
- phone, feature 13
- phonograph 30–31
- picture, cloud 227
- pipelining 51
- pixel 29, 194, 221, 286
- plain text 124, 189
- platform 106, 286
- plug-in 286
- PNG compression 74, 158, 286
- polynomial 71
- Pony Express 142
- popup blocker 194
- port 176, 189
- Portable Document Format (PDF) 116, 213, 286
- PowerPC processor 117
- powers of 10, table of 36
- powers of 2 and 10, table of 36, 281
- powers of 2, table of 36
- Pride and Prejudice* 17, 135, 158
- printers, HP 183
- Privacy Badger extension 206
- privacy policy 218, 258
- privacy, email 230
- private browsing 219
- processor family, x86 117
- processor, AMD 52
- ARM 52, 77, 117, 282
- Intel 14, 51, 76, 117
- PowerPC 117
- program library 86, 123, 128, 140, 285
- program testing 49, 65, 88, 128
- program vs algorithm 75
- programming language, Ada 9
- assembly 76
- BASIC 80
- C 80–81, 83, 95, 116, 193
- C# 96
- C++ 80–81, 83, 95, 116
- COBOL 79
- F# 96
- Fortran 79
- Go 96
- high-level 77
- Java 81, 121, 193
- JavaScript 82, 121, 193, 228, 285
- Objective-C 81
- Python 83, 122
- Rust 96
- Scratch 121
- Swift 96, 121
- Visual Basic 80
- programming style 82
- promiscuous mode 150
- `prompt` function 124–125, 129
- property, intellectual 89, 284
- protocol 163, 166, 174, 286
- protocol layers diagram 174
- protocol, application-level 174
- Bitcoin 181, 201, 261, 282
- BitTorrent 181, 282
- DHCP 166
- HTTP 177, 187, 284
- HTTPS 177, 189, 203, 206, 222, 255, 284
- IMAP 180
- IPv4 165
- IPv6 165
- SMTP 178, 180

- proximity card 153
- public records 225
- public-key algorithm, RSA 254, 286
- public-key cryptography 253
- Punycode encoding 170
- Python for loop 133
- Python import 133
- Python list 133
- Python matplotlib 133
- Python programming language 83, 122
- QR code 161
- quadratic algorithm 68–69, 286
- quetto 209
- Queen Elizabeth I 250
- Queen Elizabeth II 188
- query logs, AOL 226
- quetta 209, 281
- Quicksort algorithm 68, 91
- radio-frequency identification (RFID) 152, 286
- RAID file system 114
- random access memory (RAM) 15, 43, 286
- random vs sequential access 15
- range, Ethernet 151
- range, wireless 151
- ransomware 61, 201, 286
- RBG encoding 40
- recipe as algorithm 63
- recycle bin 112
- Red Hat 96
- Reddit 211, 227
- registrar, domain name 167, 170, 286
- reinforcement learning 240, 243, 286
- repertoire, instruction 44, 51
- representation, digital 32
- Request for Comments (RFC) 167
- reverse DNS lookup 168
- reverse Turing test 56
- RGB color representation 30, 40, 286
- RIAA 181
- right to repair 183
- Ring, Amazon 246, 270
- Ritchie, Dennis 80, 100
- Rivest, Ronald 254, 256
- RLHF 243
- Roku 221
- ronna 209, 281
- ronto 209
- root name server 169
- rot13 encryption algorithm 250
- router *see* gateway
- router vs gateway 164
- router, wireless 169, 288
- routing 166, 171
- Rowe, Mike 170
- RSA breach 205
- RSA factoring challenge 254
- RSA public-key algorithm 254, 286
- Rust programming language 96
- Safari browser 116, 123, 188, 204
- safe harbor, DMCA 182
- scareware 201
- Schneier, Bruce 6, 207, 270
- Scratch programming language 121
- script kiddy 196
- script tag 123, 129, 219
- SD card file system 113
- search algorithm, binary 66, 282
- search engine diagram 212
- search engine, AltaVista 212
 - Bing 74, 189
 - DuckDuckGo 215
 - Google 74
- secondary storage 16
- secrecy, forward 256
- secret-key cryptography 252
- Section 230 145
- Secure Digital (SD) card 19
- SecurID breach 205
- security by obscurity 252
- security laws, Microsoft 194
- selection sort algorithm 67
- self-driving car 4, 60, 233, 239, 247
- semantic error 79
- semantics 122
- sentiment analysis 240
- sequential vs random access 15
- SHA-1, SHA-2, SHA-3 message digest algorithm 256, 286
- SHA-256 257
- Shamir, Adi 254
- Shannon, Claude 27
- Shockley, William 20
- shogi (game) 240
- SI International System of Units 209, 281
- SI units, table of 281
- Signal 204
- Silicon Valley 21
- Silicon Valley* TV series 160
- Silk Road 263
- SIM (Subscriber Identity Module) 154, 201
- SIM-swapping attack 201
- Simon, Herbert A. 233
- simulator, Toy computer 45, 49, 129
- Siri 233
- slide rule 7
- smart quotes 123
- smartphone 216, 287
- SMTP protocol 178, 180
- Snapchat 258
- sniffing 150, 203
- Snowden, Edward xii, 3, 204, 224, 259–260
- social engineering attack 197, 199, 287
- social graph 224
- software components 85
- Software Development Kit (SDK) 86, 93, 286
- software layers 174
- solid state drive (SSD) 16, 110, 112, 287
- sort times, table of 70
- sound wave diagram 31
- source code 95, 194, 283, 287
- spam 197
- spear phishing attack 199, 286
- spectrum allocation 152, 162
- spectrum auction 152
- speech recognition 233
- speed of light 157
- speedtest.net 149
- Spotify 17, 30, 160, 181
- spyware 200, 287
- SQL injection attack 202
- SSD, wear leveling of 110, 112
- SSH (Secure Shell) 178
- Stallman, Richard 95
- standard, 802.11 wireless 152, 157, 162, 282
 - AES encryption 252, 282
 - ASCII 34, 40, 57, 95, 158, 178, 282
 - DES encryption 252, 283
 - Unicode 34, 39, 57, 95, 287
- standards 94, 143, 287
- statement, assignment 125–126
- statement, if-else 127
- stingray 155, 287
- Stonehenge 7
- stored-program computer 15, 55
- streaming 54
- Street View, Google 239
- Stroustrup, Bjarne 80
- Stuxnet worm 61, 196
- subroutine 85
- subscription model 93, 228
- Sun Microsystems 81, 94, 193
- Sun vs Microsoft 193
- supercomputer 54
- supervised learning 236, 287
- supply-chain attack 61
- surveillance capitalism 3, 249
- surveillance, Flock 270
- swapping 101
- Sweeney, Latanya 226
- Swift programming language 96, 121
- syntax 122
- syntax error 78
- system call 106, 287
- system, embedded 59, 81
- table of ASCII characters 34
- table of ASCII with parity 161
- table of powers of 10 36
- table of powers of 2 36
- table of powers of 2 and 10 36, 281
- table of SI units 281
- table of sort times 70
- table, binary addition 39
- table, binary multiplication 39
- tablet 11, 18
- tag manager, Google 219
- tag, *form* 191
 - HTML 189
 - img 190
 - script 123, 129, 219
- TAILS operating system 261
- Taiwan Semiconductor Manufacturing Company (TSMC) 23
- Takeout, Google 206
- Tanenbaum, Andrew 94
- Task Manager, Windows 101
- tax form as algorithm 63
- taxi dataset, New York City 226
- TCP acknowledgment 176
- TCP port 176, 189
- TCP segment diagram 176
- TCP/IP 166, 174, 181
- TCP/IP connection diagram 177
- telegraph* (Encyclopedia Britannica) 141
- television, digital 162
- Teller, Edward 35
- Telnet 178, 184
- telnet command 178
- tera 17, 281
- Tesla Autopilot 60, 93
- testing, program 49, 65, 88, 128
- tethering 157
- text concatenation 125, 127
- TextEdit 124
- third-party cookie 197, 206
- Thompson, Ken 100
- TikTok xii, 145, 211, 227
- Time Machine, Apple 206
- time to live (TTL) 175
- Tokyo in Unicode 39–40
- top-level domain 167
- top500.org 54
- Tor browser 207, 259
- Tor Project 259

- Torvalds, Linus 103
- Toy computer 44, 76
- Toy computer simulator 45, 49, 129
- Toy program in memory diagram 49
- Toyota 217
- traceroute command 172
- tracking 216
- tracking cookies 192, 217
- tracking, JavaScript 219
- trade secret 89
- trademark 89
- tradeoff 51, 83, 154, 160, 228, 268
- training set 236
- transformer architecture 243, 287
- transistor 10, 20, 91
- Transmission Control Protocol (TCP) 174, 287
- transparency report, Amazon 230
 - Facebook 230
 - Google 230
- TransUnion breach 202
- trash folder 112
- Traveling Salesman Problem 71
- Trojan horse 196, 287
- troll 287
- troll, patent 91
- Trump, Donald xii, xiv, 263
- Tukey, John 27, 35
- Turing Award 56, 68, 72, 76, 79, 100, 151, 177, 188, 235, 239, 254
- Turing machine 55–56, 287
- Turing test 55, 241
- Turing test, reverse 56
- Turing, Alan 55, 251
- Tuvalu, .tv 167
- two-factor authentication 205
- typo-squatting 199, 287
- uBlock Origin extension 206, 231
- Ukraine cyber attack 61
- Unicode domain name 168
- Unicode standard 34, 39, 57, 95, 287
- Unicode, Tokyo in 39–40
- unifers.ai 221
- United Healthcare 61
- UnitedHealthcare breach 61, 202
- units, table of SI 281
- UNIVAC 80
- universal computer 48
- Universal Serial Bus (USB) 18, 287
- universal Turing machine 55
- Unix operating system 100–101, 103, 115, 117, 287
- unsupervised learning 237, 287
- URL 187, 189, 287
- URL encoding 189
- USB file system 113
- USB virus 196
- User Datagram Protocol (UDP) 174
- UTF-8 encoding 58, 287
- vacuum tube 10, 20
- van Rossum, Guido 83
- variable declaration 124, 126, 283, 287
- VHS vs Betamax 94
- vibe coding 245, 287
- virtual currency 263
- virtual machine 105, 116, 129, 136, 140, 193, 287
- virtual machine license 105
- virtual memory 101, 140, 287
- virtual operating system 104
- virtual operating system diagram 104
- virtual operating system, Parallels 104
- VirtualBox 104
- VMware 104
- virtual private network (VPN) 203, 287
- VirtualBox virtual operating system 104
- virus vs worm 195, 287
- virus, Anna Kournikova 195–196
- floppy disk 196
- USB 196
- Visual Basic 195–196
- Visual Basic interpreter 196
- Visual Basic programming language 80
- Visual Basic virus 195–196
- VMware virtual operating system 104
- voice over IP (VoIP) 145, 174, 287
- von Neumann architecture 13, 24
- von Neumann, John 11, 13, 41, 43–44, 51, 53
- voting machines, computers in 61
- voting, Internet 61
- VPN breach 204
- W3Schools 122
- wafer, integrated circuit 21
- walled garden 269, 272, 287
- Waymo 60
- Waze 153
- wear leveling of SSD 110, 112
- web beacon 194, 217, 288
- web server, Apache 96
- web server, NGINX 96
- well known port 176
- WhatsApp 204, 223, 271
- while loop 126
- whois command 168
- Whorf, Benjamin 84
- Wi-Fi 108, 152, 162, 288
- Wikipedia 96
- Wilkes, Maurice 75–76
- Williamson, Malcolm 254
- Windows File Recovery program 112
- Windows operating system 11, 40, 100, 103, 105, 107, 117, 195–196, 198, 228
- Windows Task Manager 101
- wine measures, English 41
- Wintel 117
- wireless base station 151, 282
- wireless encryption (WEP, WPA) 162
- wireless range 151
- wireless router 169, 288
- wireless standard, 802.11 152, 157, 162, 282
- Wireshark 150
- World Wide Web Consortium (W3C) 188
- worm vs virus 195, 288
- worm, Morris 195
- worm, Stuxnet 61, 196
- x86 processor family 117
- Xbox SDK 86
- XCode IDE 86
- Xerox PARC 149
- Yahoo 175, 230, 260
- Yamamoto, Isoroku 250
- yopmail.com 207
- yotta 209, 281
- YouTube 211, 224, 227
- zero-day attack 88
- zetta 209, 211, 281
- Zillow 225, 237
- ZIP compression 74, 158
- zombie 200
- Zoom 144, 204
- Zuboff, Shoshana 3, 249
- Zuckerberg, Mark 184