

INDEX

A

- absolute addressing, 129
- abstractions, 273–274, 433–434, 435
- accumulator register, 103–104
- active high and low, 72
- active pull-up switch, 58
- ADC (analog-to-digital) converters, 162–165
- adders, 60–63
- addition, 8–10, 10–14
- additive color system, 28, 173
- addressing
 - and I/O devices, 96–97
 - memory, 79–81
 - modes, 104–105
 - with pointers, 184–185
 - relative and absolute, 128–130
- Adleman, Leonard, 368
- advisory locks, 339–340
- Aho, Alfred, 228, 438
- AI. *See* artificial intelligence (AI)
- AJAX (Asynchronous JavaScript and XML), 252
- algorithm efficiency vs. performance, 215
- aliasing, 170, 180
- ALU (arithmetic logic unit), 97–99
- American Standard Code for Information Interchange (ASCII), 22–24, 213
- Ampère, André-Marie, 44
- amplitude, 155, 165
- analog comparators, 163, 164
- analog devices
 - characteristics, 35–36, 37–38
 - and transfer functions, 38–40
- analog-to-digital (ADC) converters, 162–165
- Anathem* (Stephenson), xxxv
- ancestor node, 243
- AND
 - logic gates, 53–54, 59
 - operation, 4–5, 5–6, 9
 - in plumbing example, 42–43
 - with relays, 49
- Andreesen, Mark, 251
- Android operating system, 376
- animation, 29–30
- anodes, 50
- anonymized data mapping, 410–411
- anonymous functions, 266
- APIs (application program interfaces), 433–436
- Apple, 417, 433
- application program interfaces (APIs), 433–436
- application vs. system programming, 259, 282
- approximations and shortcuts
 - CORDIC algorithm, 313–318
 - efficiency goals, 283
 - integer methods, 290–301
 - of power series, 313
 - quantization, 323–333
 - randomness, 318, 322–323
 - recursive subdivision, 301–312
 - table lookups, 284–290
- Arduino, 119
- arithmetic logic unit (ALU), 97–99
- Armel processors, 142
- ARPANET, 157
- arrays, 185–187
- artificial intelligence (AI)
 - concepts, 388
 - development, 385–386, 407
 - and neural networks, 402
 - self-driving ketchup bottle example, 407–409
- ASCII (American Standard Code for Information Interchange), 22–24, 213
- Asente, Paul, 439
- assemblers, 218, 233–234
- assembly language, 217–218
- asynchronous counters, 77

asynchronous functions and promises, 346–353
Asynchronous JavaScript and XML (AJAX), 252
AT&T, 155
atomic operations, 339, 342, 343
attack surfaces, 355, 373–374
audio
 amplifier transfer function, 39–40
 differential signaling applications, 57
 digital representation, 165–173
 frame layout, 210
audio filters, 168–169
authentication, 356, 358–359, 361–362, 370
authorization, 361
autodialers, 355
autoincrement/autodecrement modes, 114
axon terminals, 401

B

B-trees, 205
Babbage, Charles, 35
back-EMF effect, 48
backdoors and security, 356, 368, 373–374
backpropagation, 404
Backus, John, 222
Backus-Naur form (BNF), 222–223
 examples, 226–227
bandpass filters, 168
bandwidth, 156
Barlow, John Perry, 357
barrel shifters, 100
base-2 system, 6
Base64 encoding, 26–27
bash shell, 437–438
BASIC, 219–220
batch processing, 176
Battle of Midway code breaking, 366–367
Baud rate, 154
Baudot, Émile, 154
Bayer, Bryce, 325
Bayer matrix, 325–326
Bayer, Rudolf, 205
Bayes' theorem, 389–390
Bayes, Thomas, 389
BCD (binary-coded decimal) system, 18
Bechtolsheim, Andy, 439

Bell, Alexander Graham, 56
bell curve, 390–391
Bell Telephone Laboratories, 150, 179, 209, 220, 225, 416, 442
Bentley, Jon, 228
Berners-Lee, Sir Tim, 159, 239–240
Berryman, Jeff, 134
big data, 387, 409–412
binary-coded decimal (BCD) system, 18
binary, defined, 3
binary numbers
 addition with, 8–10
 coded as decimals, 18
 context notation, 20
 as integers, 6–8
 as negative numbers, 10–14
 octal and hexadecimal forms, 18–20
 as real numbers, 14–18
binary thresholds, 41
binary trees, 199–203
binning, 71
bipolar junction transistors (BJTs), 51
bison program, 226
bit density, 87
bitmaps, 187–188, 204, 312
bits
 as binary numbers, 6, 8
 defined, 3
 groupings, 20–22
 overflow, 10
 page table control, 131
 as right choice for technology, 33–34, 40–41
BJTs (bipolar junction transistors), 51
Blaze, Matt, 373–374
blits (terminals), 209
block storage devices
 addressing, 203–204
 hardware, 85–88
blockchain, 371
blocking mode, 341
Bluetooth, 158, 352–353
Boole, George, 4, 386
Boolean algebra, 4–5
booting, 218
bootstrap, defined, 218
Bourne, Stephen, 438
branch prediction, 135
branching instructions, 105–106
Bray, John, 29
breadth-first traversal, 123
break statement (C), 195, 196

- Bresenham, Jack, 294–295
 - browsers. *See* web browsers
 - buffer overflows, 275, 374–375
 - buffers
 - in logic gates, 53
 - program, 270–273, 274
 - raster frame, 311
 - bugs. *See also* errors
 - buffer overflow, 275
 - likelihood of, 282
 - reporting and tracking, 441
 - term origin, 50
 - build tools, 421
 - Burks, Arthur, 125
 - buses, 80, 94–95, 96–97
 - Bush, Vannevar, 159
 - button circuits, 144–146, 147–148
 - bytes, defined, 21
- C**
- C programming language
 - brief overview, 114, 220
 - compiler, 268
 - input and output, 274–275
 - optimized code examples, 235
 - primitive data types, 184–189
 - runtime libraries, 275–276
 - sorting functions, 213
 - unions, 190
 - C++ language concepts, 211–212
 - CA (certificate authorities), 370–371
 - cache management, 134–135
 - calculator program examples, 226–227, 229–230
 - Canny, John, 398
 - canvas, 255, 290–291
 - card reader technologies, 84, 85
 - career success
 - decision-making, 427–428
 - estimating and scheduling, 426–427
 - job/career vs. calling, 430
 - and open source projects, 442–443
 - working with people, 428–429
 - and workplace culture, 429–430
 - Carpenter, Loren, 322–323, 429
 - carrier waves, 155–156
 - Cartesian coordinate mapping, 291, 301
 - Cascading Style Sheets (CSS), 244–248, 267
 - cat vs. meatloaf image example, 388, 391–393, 396–400
 - cathode ray tube (CRT) terminals, 177–179
 - cathodes, 50
 - CDs, 87–88, 170
 - cel animation, 29–30
 - cell phone programs, 425
 - cell phone systems
 - security exposures, 361–362, 373, 376
 - surveillance, 359
 - central processing unit (CPU), 97–102, 118–119
 - certificate authorities (CAs), 370–371
 - chaining code, 348–349
 - Chang Xiao, 363
 - Changxi Zheng, 363
 - characters
 - classification, 288–290
 - control, 23–24
 - defined, 22
 - graphics display, 311–312
 - and language variations, 439
 - numbers as, 25–27
 - sorting, 213
 - and steganography, 362–363
 - checksum method, 89
 - chem (language), 228
 - Cheng Zhang, 363
 - Cheriton, David, 439
 - child nodes, 243
 - chips. *See also* specific types
 - design, 90, 119, 127, 376
 - economics, 154
 - invention of, 52
 - chord construction, 166–167, 168
 - ciphers
 - complex, 366–367
 - one-time pads, 367
 - substitution, 363–365
 - transposition, 365–366
 - ciphertext and cleartext messages, 363
 - circuit-switched networks, 157
 - circuits, 44, 47
 - circular buffers, 272–273
 - CISC. *See* complicated instruction set computers (CISC)
 - class attribute (CSS), 267

- classifiers
 - and artificial intelligence, 387–388
 - feature recognition, 399–400
 - ketchup bottle example, 406
 - naive Bayes classifier, 389–390
 - neural networks as, 405
- cleartext and ciphertext messages, 363
- Clipper chip, 374
- clocks, 71, 77
- cloud computing, 424
- clusters, 87, 203
- CMOS (complementary metal oxide semiconductor), 52
- CMRR (common-mode rejection ratio), 56
- coalescing nodes, 307
- code
 - data as, 382–384
 - maintainable, 441–442
 - portable, 439–440
 - refactoring, 441
 - self-modifying, 407
 - source control and distribution, 440
 - testing, 440–441
 - third-party, 376–378
 - writing vs. reusing, 436
- code breaking, 366–367
- code (machine language) generators, 233–234
- codecs, 172
- coding tools, 437–439
- collisions
 - Ethernet, 158
 - half-duplex, 154
 - in hash tables, 214–215
- color displays, 173–174, 181
- color gradients, 296–297
- color representations, 27–30, 190–191
- Colossus: The Forbin Project* (film), 409
- combinatorial logic, 53
- command and control messages, 358
- command interpreters, 437–438
- command line interface, 268
- common-mode rejection ratio (CMRR), 56
- communications security, 356–357
- compact discs, 87–88, 170
- compare and swap instruction, 342
- compilers
 - defined, 219
 - execution, 232–234
 - vs. interpreters, 228–229
- complicated instruction set computers (CISC), 113, 114
- compositing, 30
- compound data types
 - doubly linked lists, 198–199
 - and memory allocation, 195–198
 - singly linked lists, 191–195
 - suites/structures, 189–190
- computer animation, 30
- computer architecture
 - basic elements, 118–119
 - defined, 117–118
- computer vision libraries, 399
- computers
 - brief history, 416–418
 - languages, 1–3, 217–218, 219–220
 - major components of, 93–94, 109–113
 - resource usage efficiencies, 283
 - stored-program, 101
- computing devices
 - analog vs. digital, 35–36, 37–38
 - mechanical, 34–35
- Concurrent Versioning System (CVS), 440
- condition code instructions, 105
- condition code register, 10, 98
- conductors, 43
- constants, 221
- constructive solid geometry technique, 304–310
- containers, 422
- context
 - switching, 269–270
 - and symbols, 2, 3
- continuous values, 36, 37–38
- control characters, 23–24
- control unit. *See* execution unit
- conversion tables, 284
- convolution kernels, 392–393
- convolutional neural network, 405
- Conway, Lynn, 90
- cooked buffer mode, 271
- Coordinate Rotation Digital Computer (CORDIC) algorithm, 313–318

- coprocessors, 135–136
 - CORDIC (Coordinate Rotation Digital Computer) algorithm, 313–318
 - core memory, 82–83
 - core rope memory, 84
 - counters, 77–78
 - CPU. *See* central processing unit (CPU)
 - CRCs (cyclic redundancy checks), 89
 - Creative Commons, 420
 - crosstalk effect, 38
 - CRT (cathode ray tube) terminals, 177–179
 - cryptography. *See also* ciphers; encryption/decryption
 - blockchain, 371
 - concepts, 357, 362
 - and digital signatures, 370
 - hash functions, 369–370
 - and password management, 371–372
 - public key, 368
 - steganography, 362–363
 - Cryptonomicon* (Stephenson), 357
 - crystals, 70–71
 - CSS. *See* Cascading Style Sheets (CSS)
 - CSS selectors, 245–247
 - Curie, Jacques and Pierre, 70
 - current (I), 44
 - cutoff regions, 41
 - CVS (Concurrent Versioning System), 440
 - cyclic redundancy checks (CRCs), 89
- D**
- D/C converters, 161–162
 - D flip-flops, 74–76
 - DAC converters, 161–162
 - DAGs. *See* directed acyclic graphs (DAGs)
 - dark adaptation, 29
 - Dark Star* (film), 388
 - data. *See also* big data; encryption/decryption; personal data
 - as code, 382–384
 - compression, 172
 - copying and moving, 206–211
 - machine training, 387, 405, 406
 - protection of, 352, 354–355, 359–360, 378–379
 - static and dynamic, 136
 - data centers, 424
 - Data Encryption Standard (DES)
 - cracker, 216
 - data mining, 387
 - data paths, 109–110
 - data structures
 - linear vs. hierarchical, 199
 - and performance, 183–184
 - spatial, 123
 - data types. *See* compound data types; primitive data types
 - database management systems (DBMS), 205
 - databases, 204–206, 216
 - datagrams, 158–159
 - date-time structure, 189–190
 - DBMS (database management systems), 205
 - DDoS (distributed denial of service)
 - attacks, 357
 - De Morgan's law, 5–6, 54
 - deadlocks, 341–342
 - debouncing, 144–146
 - debugging practices. *See also* error checking, 378
 - DEC. *See* Digital Equipment Corporation (DEC)
 - decimal number system, 6, 18, 20
 - decision criteria, 37–38, 54–55
 - decision-making skills, 427–428
 - decoders, 63–64
 - demand paging, 132
 - demodulation, 156
 - demultiplexers (dmux), 64–65
 - dendrites, 401
 - denial of service (DDoS) attacks, 357
 - depth-first traversal, 123, 244
 - DES (Data Encryption Standard)
 - cracker, 216
 - descendant node, 243
 - desktop publishing, 254
 - detents, 38
 - development methodologies, 430–431
 - device drivers, 269, 270–273
 - diff program, 440
 - difference engine, 35
 - differential signaling, 55–57
 - Diffie–Hellman Key Exchange, 368
 - Diffie, Whitfield, 368
 - digital audio, 165–173
 - digital camera technology, 38–39, 82, 325

- digital devices, characteristics, 35–36, 37–38
 - Digital Equipment Corporation (DEC) systems
 - PDP-10, 407
 - PDP-11, 21, 114
 - vs. UNIX, 434
 - digital images, 173–176
 - digital signal processors (DSP), 15
 - digital signatures, 370
 - digital-to-analog (DAC, D/C) converters, 161–162
 - digits, 3, 6
 - digraphs, 365
 - diodes, defined, 142
 - direct addressing mode, 104
 - direct memory access (DMA) units, 136
 - directed acyclic graphs (DAGs), 123, 229, 242–243
 - directories, 204
 - discrete values, 36, 37–38
 - disk drives, 85–87, 95, 203–204
 - display lists, 179, 181
 - displaying
 - characters, 312
 - ellipses, 298–300
 - gasket example, 304–310
 - images, 173–176
 - polynomial shapes, 301
 - spirals, 301–304
 - straight lines, 292–296
 - displays
 - flat-screen, 178
 - LED, 146–148
 - distortion, 39–40, 169
 - distributed denial of service (DDoS) attacks, 357–358
 - dithering, 325–333
 - division by zero, 18
 - DMA (direct memory access) units, 136
 - DNS (Domain Name System), 159
 - Document Object Model (DOM)
 - brief history, 344
 - and CSS selectors, 245
 - manipulation, 252–253
 - structure, 242–244
 - Document Type Definition (DTD), 250
 - documentation, 432, 442–443
 - dog-whistle marketing, 363
 - DOM. *See* Document Object Model (DOM)
 - Domain Name System (DNS), 159
 - domain-specific languages, 228
 - doping, 51, 376
 - DoS (denial of service) attacks, 357
 - double-pole, double-pole (DPDT) switches, 46
 - double-precision numbers, 17–18
 - doubly linked lists, 198–199
 - DRAM (Dynamic RAM), 82, 134
 - drawing. *See* graphics
 - DSPs (digital signal processors), 15
 - Duff, Tom, 30, 208
 - Duff's Device, 208–209
 - duty cycle, 148–149
 - DVDs, 87–88
 - dynamic data, 136
 - dynamic memory (DRAM), 82, 134
 - dynamic memory management, 195–198, 379–381
- ## E
- EBCDIC (Extended Binary-coded Decimal Interchange Code), 22
 - Eccles, William, 76
 - echoing, 270, 271
 - edges
 - detection, 393–398
 - logic transition, 74
 - tracking with hysteresis, 398–399
 - EEC (error checking and correcting) chips, 89
 - EEPROM (electrically erasable programmable read-only memory), 85, 88
 - EFF. *See* Electronic Frontier Foundation (EFF)
 - electricity
 - plumbing analogy, 41–44
 - switches and circuits, 44–47
 - electromagnetic deflection, 178
 - electromagnets, 48
 - Electronic Frontier Foundation (EFF), 216, 360, 363
 - electrostatic deflection, 177–178
 - ELF (Executable and Linkable Format), 137
 - encoding
 - bit patterns, 24–25
 - color and light, 30, 149–151
 - defined, 2

encryption/decryption. *See also*
 cryptography
 asymmetric, 368
 cipher types, 363–367
 concepts, 356–357
 forward secrecy, 369
 key exchange, 367–368
 one-time pads, 367
 Public Key Infrastructure (PKI),
 370–371
 standards development, 360
 end-around carry, 12
 endianness, 96, 184
 Engelbart, Douglas, 181
 entity references, 241
 entropy harvesting, 375–376
 EPROM (erasable programmable read-
 only memory), 85
 equation notations, 125
 error checking. *See also* debugging
 practices
 in memory allocation, 276–277
 practices, 373
 error checking and correcting (ECC)
 chips, 89
 error message output, 274–275, 277
 error propagation, 329–333
 errors. *See also* bugs
 dynamic memory allocation, 196,
 197–198
 and logic circuit design, 71
 in memory, 88–89
 estimation skills, 426
 Ethernet, 158, 341
 event handlers, 254, 266
 event loops, 343
 event queues, 344
 events, 128
 exception handling, 276
 exclusive-OR (XOR) operation, 4–5,
 9, 53
 Executable and Linkable Format
 (ELF), 137
 execution unit, 100–102
 expert systems, 407
 exponent, 16, 17–18
 Extended Binary-coded Decimal
 Interchange Code
 (EBCDIC), 22
 eXtensible Markup Language (XML),
 239, 248–251

F

Fantasia (film), xxx
 feedback, 70, 72
 feedforward networks, 403
 Feldman, Gary, 395
 Feldman, Stuart, 414
 Fender Bluetooth guitar exposure,
 352–353
 fetch-execute cycle, 109, 111
 FETs (field effect transistors), 51–52
 Fibonacci sequence program code,
 107–108, 218, 219, 220
 field effect transistors (FET), 51–52
 field-programmable gate arrays
 (FPGA), 90, 337
 FIFO (first-in, first-out) applications,
 162, 270
 file descriptors, 271, 274, 435
 file pointers, 274
 filenames, 203, 271–272
 files
 as locks, 343
 treatments of, 434
 filesystems, 204
 filters, voltage frequency, 168
 finite impulse response (FIR) filters,
 145–146
 firmware, 90
 first-in, first-out (FIFO) applications,
 162, 270
 fixed-point numbers, 14–15
 FLAC (Free Lossless Audio Codec),
 172–173
 flash converters, 163–164
 flash memory, 88, 382
 Fleming, Sir John Ambrose, 50
 flex program, 225
 flight computers, 35
 flip-flops, 74–76
 floating-point arithmetic, 100, 290, 294
 floating-point numbers, 15–18,
 221–225
 floppy disks, 87
 flowcharts, 125–126
 Floyd, Robert, 330
 Floyd-Steinberg dithering algorithm,
 330–331
 FM stereo, 171–172
 folding. *See* aliasing
 forth (programming language), 124
 FORTRAN, 212–213, 219–220, 226

forward secrecy, 369
 Fourier, Jean-Baptiste Joseph, 167
 Fourier transform, 167
 Fournier, Alan, 322
 FPGAs (field-programmable gate arrays), 90, 337
 fprintf function (C), 277
 fractals, 319–323
 fragmentation of memory, 196
 frame buffers, 311
 free function (C), 195–196, 197, 379–381
 Free Lossless Audio Codec (FLAC), 172–173
 free space tracking, 204
 frequencies

- defined, 155
- filtering, 168–170

 frequency shift keying (FSK), 155–156
 Friedman, Elizebeth Smith, 363
 fskc program, 204
 FSK (frequency shift keying), 155–156
 full adder, 61
 full-duplex connection, 154
 function calls, 120–121
 functions

- and libraries, 137–138
- vs. macros, 290
- trapdoor, 368

 Fussell, Don, 322
 fuzzing, 375

G

garbage collection, 197–198, 381–382
 gate arrays, 90
 gated latches, 73–74
 Gates, Bill, 358
 gates. *See* logic gates
 Gauss, Johann Carl Friedrich, 390
 Gaussian blur, 391–393
 Gaussian distribution, 390–391, 394
 General Data Protection Regulation (GDPR), 379
 General Electric, 416
 general purpose computers, 15
 Generalized Markup Language (GML), 239
 genetic algorithms, 407–409
 Geschke, Chuck, 254
Ghostbusters (film), 58
 gigabytes, defined, 21
 Gilmore, John, 418
 Git, 440
 GLANCE G terminals, 179
 glass ttys, 178
 glibc function, 436
 GML (Generalized Markup Language), 239
 GNU project, 415, 418, 421
 Goldberg, Adele, 158
 Gosling, James, 422, 443
 governments and privacy, 359–361, 373–374
 GPUs (graphics processing units), 114–115
 gradient descent algorithm, 404
 gradients

- color, 296–297
- in edge detection, 395–396

 granularity of locks, 340–341
 graphic equalizers, 168
 graphical user interface (GUI), 268, 338
 graphics

- canvas, 290–292
- color gradients, 296–297
- constructive solid geometry, 304–311
- drawing curves, 298–304
- drawing straight lines, 292–296 and randomness, 322–323
- shifting and masking, 311–312

 graphics processing units (GPUs), 114–115
 graphics rotations, 291
 graphics terminals, 177–178
 Grateful Dead recordings, 57
 Gray code, 150
 Gray, Frank, 150
 grayscale

- displays, 178
- quantization, 323–333

 Greenblatt, Richard, 407
 grep utility (UNIX), 224
 “Guess the animal” program, 260–262, 263–266, 276–282
 GUI (graphical user interface), 268, 338

H

half adder, 61
 half-duplex connection, 154
 halftone printing, 325
 Hamming codes, 89

Hamming, Richard, 89
 Hammond B-3 organ, 168
 hard drives, 87
 hardware. *See also* logic gates
 and code optimizers, 236
 computing technology
 components, 47–52
 design issues, 36–38, 60, 119
 electrical switches and circuits,
 44–47
 logic gates, 53–60
 manufacturing issues, 71
 and software, 90
 hardware exception handling, 133
 Harvard architecture, 118
 hash code checking, 89
 hash functions, 213–215, 369–370
 heap, 136–137
 Hellman, Martin, 368
 Henkel-Wallace, DV, 418
 Hennessey, John, 113
 Hertz, Heinrich, 155
 Hertz (Hz), 155
 Hewlett Packard (HP)
 microcode implementations, 113
 reverse Polish notation
 calculators, 125
 hex triplets, 30
 hexadecimal representation, 19–20
 hi-Z (hi-impedance) state, 60
 Hibernate tool (Java), 423
 hidden layers, 403
 hierarchical data structures, 199–203
 hierarchical filesystems, 204
 high pass filters, 168
 Hilbert curve, 320
 Hilbert, David, 320
 Hinton, Geoffrey, 404
A History of Personal Workstations
 (Goldberg), 158
 hold time, 76
 Hollerith cards, 83–84
 Hollerith, Herman, 83
 Honeywell computers, 20
 Hopper, Grace, 50
 horizontal partitioning, 216
 host (URL), 240
 Hough transform, 399
 HTML. *See* HyperText Markup
 Language (HTML)
 HTML5, 255
 HTTP. *See* HyperText Transfer Protocol
 (HTTP)
 human interface devices, 176–181
 humans. *See also* programmers
 hearing, 170–171, 172–173
 language, 1–3
 nervous system, 387
 thought processing, 386–387
 vision, 29, 147, 174, 291, 393
 Hurd, Earl, 29
 hypertext, 159, 240
 HyperText Markup Language (HTML)
 documents, 240–242, 243
 elements and attributes, 241–242,
 245–248
 evolution, 160, 238, 239
 HyperText Transfer Protocol
 (HTTP), 159
 hypervisor, 425
 hysteresis applications, 54–55, 398–399

I

I/O. *See also* input and output
 vectored, 210–211
 I/O devices
 computer access to, 96–97, 141–142
 mediation, 268–269
 on-chip, 127
 sharing, 337
 I/O ports, 142–144
 IBM
 Hollerith cards, 83, 84
 Selectric terminals, 177
 IDE interface, 152, 154
 IDEs (integrated development
 environments), 439
 IEEE floating-point numbers, 17–18
 IF statements, 212–213
 image processing and recognition
 edge detection, 393–398
 edge tracking with hysteresis,
 398–399
 feature extraction, 399–400
 Gaussian blur, 390–393
 nonmaximum suppression, 398
 ImageMagick, 436
 images. *See also* graphics
 digital representation, 173–176
 texture mapping, 285–288
 immediate addressing mode, 105
 impedance (Z), 60

- index register, 129
- indices
 - array, 185
 - database, 206
 - hash table, 214–215
- indirect address registers, 110
- indirect addressing
 - and linked lists, 193–194
 - mode, 104
- indirect blocks, 204
- infix notation, 125, 227
- inline styles, 267
- inodes, 203–204
- input and output
 - computer access to, 96–97
 - device drivers, 268–269, 270–273
 - in UNIX file abstraction, 435
- inputs. *See also* noise
 - error-checking, 373, 374
 - and transfer functions, 39–40
- Institute of Electrical and Electronic Engineers (IEEE), 17
- instruction register, 109
- instructions. *See also* code
 - addressing modes, 104–105
 - as bit patterns, 101
 - branching, 105–106
 - condition codes, 105
 - data as, 382–384, 387–388
 - layouts, 102–104, 106–107
- insulators, 43
- integer methods
 - in CORDIC algorithm, 313–318
 - drawing curves, 298–300
 - drawing gradients, 296–297
 - drawing straight lines, 295–296
 - and performance, 290
 - with polynomials, 301
- integer representations, 6–8
- integrated circuits. *See also* chips; logic gates, 52, 53, 100
- integrated development environments (IDEs), 439
- integrated peripherals, 127
- integrity verification, 370
- Intel, 90, 113
- interface design, 433–436
- interference, 37–38
- interior node, 243
- International Standards Organization (ISO) characters, 24

- internet
 - accessing, 158–160
 - as attack surface, 357–359
- interpreters
 - vs. compilers, 228–229
 - execution, 231–232
 - web browsers as, 237
- interrupt handlers, 129–130, 375
- interrupts, 125–128, 341
- inverters, 49, 53–54, 70, 72
- IP addresses, 159
- isochronous transfers, 156

J

- Japanese Industrial Standard (JIS)
 - characters, 24
- Java programming language, 198, 416, 422–423
- Javadoc, 443
- JavaScript language
 - and asynchronous issues, 343–346
 - function example, 120
 - and garbage collection, 198, 381–382
 - “Guess the animal” game, 264–266, 276
 - and jQuery, 254
 - and JSON, 255–256
 - promise construct, 346–350
 - as self-modifying code, 407
 - and web browser, 251–253
- JavaScript Object Notation (JSON), 255–256
- Johnson, Stephen C., 226
- Jordan, Frank, 76
- JPEG compression, 122–124, 174
- jQuery, 253–254, 345–346
- JSON (JavaScript Object Notation), 255–256

K

- Kernighan, Brian, 228, 434, 438
- ketchup bottle AI example, 406, 408–409
- key exchanges, 367–369
- keyboards, 181
- keyframes, 176
- Kilby, Jack, 52
- kilobytes, defined, 21
- Kleene, Stephen Cole, 224

kleptography, 355, 368
knife switches, 45
Knight, Tom, 407
Knuth, Donald, 435
Koch, Helge von, 319
Koch snowflake, 319

L

L-systems (Lindenmayer systems),
320–322
LAN. *See* local area networks (LAN)
Landin, Peter, 189
languages. *See also* programming
languages
 human vs. computer, 1–3, 267–268
 markup, 238–239, 248–251
Lantz, Keith, 439
large-scale integration (LSI) parts, 60
latches, 71–74
LavaRand, 376
Lawson, Harold, 184
layering video, 176
LCD (liquid crystal display), 178
leading zeros, 8
leaf nodes, 123
least recently used (LRU) algorithm, 132
least significant bit (LSB), 8
LEDs. *See* light-emitting diodes (LEDs)
Lesk, Mike, 225
lex program, 225–226
lexical analysis, 221–226
LFSR (linear feedback shift
 register), 375
libraries, 137–138, 288–289, 399, 436
LIFO (last in, first out) structure, 124
light-emitting diodes (LEDs), 59,
 142–144, 146–148, 160–161
limit registers, 124
Lindenmayer, Aristid, 320
Lindenmayer systems (L-systems),
 320–322
linear feedback shift register
 (LFSR), 375
linear region, 39–40
linked lists, 191–195, 198–199
linker programs, 137–138
<link> element (HTML), 246
links
 directory, 204
 hypertext, 240
Linux, 418, 437

liquid crystal display (LCD), 178
LISP programming language, 407
living documents, 238
load-store architecture, 113
local area networks (LAN), 156–157, 158
locality of reference, 184
lock authorities, 341
locks
 advisory, 339–340
 deadlocks, 341–342
 granularity, 340–341
 implementations, 342–343
logic gates
 concepts, 53–54
 and hardware design, 60
 output variations, 58–60
 and propagation delay, 57
 Schmitt triggers, 55, 56
logic operations
 binary addition as, 9–10
 concepts and laws, 3–6
loop-invariant optimization, 235, 288
lossless and lossy compression, 172
low pass filters, 168–169, 170
LRU (least recently used) algorithm, 132
LSB (least significant bit), 8
Łuskasiewicz, Jan, 125

M

MAC (Media Access Control)
 addresses, 158
McCarthy, John, 198, 407
McCauley, Clark, 429
McCreight, Ed, 205
machine intelligence. *See also* artificial
 intelligence (AI); big data;
 machine learning (ML), 385
machine language, 218, 233–234
machine learning (ML). *See also* image
 processing and recognition
 concepts, 386–388
 edge detection, 390–399
 feature extraction, 399–400
 naive Bayes classifier, 389–390
 and neural networks, 405
 technology trends, 385–386
 training data, 406
McIlroy, Doug, 435, 440
Macintosh API, 433
macros, 290
magnetic tape, 87

maintenance, 441–442
 malloc function (C), 195–196, 197, 379–381
 man-in-the-middle attacks, 357, 368
 Mandelbrot, Benoit, 319
 mantissa, 15–16, 17–18
 MapReduce, 216
 mark-space signaling, 153–154, 155–156
 markup languages, 238–239, 248–251
 mask-programmable ROM, 85
 masking. *See* shifting and masking
 masks

- bitmap, 187
- defined, 85
- interrupt controls, 128
- in raster graphics, 311–312

 mass storage, 85–87
 Massachusetts Institute of Technology, 407, 416
 MD5 hash function, 370
 Mead, Carver, 90
 Media Access Control (MAC)

- addresses, 158

 medium-scale integration (MSI)

- parts, 60

 megabytes, defined, 21
 memory. *See also* storage technologies

- arranging data in, 136–137
- computer access to, 94–96
- error detection and correction, 88–89
- hierarchy and performance, 133–135, 138
- organization and addressing, 79–81
- random access, 82
- read-only, 83–85
- relative addressing, 129–130
- as shared resource, 337

 memory chips, 81
 memory controller, 134
 memory management. *See also* buffer overflows

- bug prevention, 373, 374–375
- in C programming, 276–280
- dynamic allocation, 195–197, 379–381
- garbage collection, 197–198, 381–382

 memory management units (MMUs)

- design and operations, 130–132, 133
- and libraries, 138, 195

Men in Black (film), 356
 messages, command and control, 358
 metadata and security exposure, 359
 metal-oxide semiconductor field effect transistors (MOSFETs), 52
 Metcalfe, Bob, 158
 methodology vs. ideology, 430–431
 methods, C++, 211–212
 microcode, 112–113
 microcomputers, 119, 137, 375
 microprocessors, 119, 141–142
 Microsoft, 339, 355, 358, 417
 Miller, Frank, 367
 MIP mapping, 285–288
 MIT. *See* Massachusetts Institute of Technology
 MKUltra government program, 360
 ML. *See* machine learning (ML)
 MMUs. *See* memory management unit (MMUs)
 modems, 156
 modulation/demodulation, 155–156
 moiré artifacts, 328
 MOSFETs (metal-oxide semiconductor field effect transistors), 52
 most significant bit (MSB), 8
 motion compression, 176
 mouse technology, 151, 181
 MP3 frame layout, 210
 MSB (most significant bit), 8
 multicore processors, 119
 multiplexers (mux), 65–66
 multiplexing, examples, 81, 147
 multiplication, 100
 multiprocessor systems, 118–119, 216
 multitasking, 118, 133, 177, 335–336
The Mythical Man-Month: Essays on Software Engineering (Brooks), 219

N

naive Bayes classifier, 389–390
 namespaces, 249
 NaN (not a number), 18
 NAND gates, 53–54
 Napier, John, 34
 National Security Agency (NSA), 355, 368, 374
 Naur, Peter, 222
 negative logic, 5–6
 negative number representation, 10–14

Nelson, Nils Peter, 288
networking, 156–160
neural networks, 401–406
new operator, 198
nixie tubes, 63
no-execute bit, 131
Node.js, 424
nodes. *See also* trees
 adding new, 280–281
 in C programming, 276
 coalescing, 307
 leaf, 123
 lexicon, 243
noise
 and differential signaling, 55–57
 immunity, 38, 54–55
nonaligned access, 95
nonblocking mode, 341
nonmaximum suppression, 398
nonrepudiation, 370
NOR gates, 53–54, 72
normalization, of numbers, 17
NOT
 operation, 4–5, 5–6, 11
 with relays, 49
notch filters, 168
Noyce, Robert, 52
NUL terminator, 188, 189
numbers as characters, 25–27
nuxi syndrome, 96
Nyquist, Harry, 169

O

object code, 219
object-oriented programming
 concepts, 211–212
octal representation, 18–19, 20
octets, 24–25
octrees, 310–311
Ohm, Georg Simon, 44
Ohm’s law, 44
one-time pads, 367
one’s complement representation, 11–13
opcodes, 97, 98
open-collector (or open-drain)
 outputs, 58–59, 148
open source software, 377, 418,
 419–420, 436, 442, 443
OpenCV library, 399
OpenGL graphics language, 181
OpenSSL cryptography library, 377, 436

operands, 97, 221
operating systems (OS)
 context switching, 269–270
 and files, 271–272
 and I/O devices, 259–260, 268–269
 locking functionality, in, 341
 operations, 118, 128–129
 with programs vs. browsers, 273–274
 threads, 337–339
 time-sharing, 177
optical disks, 87–88
optimizers, 234–236
OR
 logic gates, 53–54
 operation, 4–5, 5–6
 in plumbing example, 43
 with relays, 49
Ørsted, Hans Christian, 47
OS. *See* operating systems (OS)
oscillators, 70–71
Ossanna, Joseph, 274
out-of-order execution, 135
outputs
 in differential signaling, 56
 of gates, 58–60
 and transfer functions, 39–40
overclocking, 71
overflow condition, 10

P

package management, 421–422
packet-switched networks, 157
packets (USB), 156
padding, 190
page fault exception, 131
page swapping, 132
page tables, 130–131
pages, 82, 130–131
The Paging Game (Berryman), 134
parallel communications, 152, 154
parallel connections, 43
parallel processing, 119
PARC. *See* Xerox Palo Alto Research
 Center (PARC)
parent node, 243
parity checking, 89
parse trees
 construction and evaluation,
 229–230, 231
 examples, 242–243
 optimizing, 234

Pascal (programming language), 220
 passive pull-ups, 59
 password exposures, 353, 354, 378
 password management, 371–372
 path (URL), 240
 pattern matching, 224–225
 Patterson, David, 113
 PCs. *See* personal computers (PCs)
 PDF (Portable Document Format),
 254–255
 Peano, Giuseppe, 319
 perceptrons, 402–403
 periodic signals, 70
 peripherals, 96, 127
 personal computers (PCs), 417, 418
 personal data
 privacy, 352, 359–361, 410–412
 and trust, 353–355, 361
 phase difference, 170
 phone security, 359, 361, 362, 373
 phones. *See* cell phone programs; cell
 phone systems
 photolithography, 51
 physical security, 355–356
 piezoelectric effect, 70
 Pike, Robert, 24, 209, 434
 pins, defined, 127
 pip (Peripheral Interchange Program,
 DEC), 434
 pixels
 in Gaussian blur, 392–393
 as image representation, 27, 173
 and MIP mapping, 286
 unions, 190–191
 in video, 175
 voxels, 310
 PKI (public key infrastructure),
 370–371
 pointers, 114, 184–185, 212
 polar coordinates, 301–304
 Polish notation, 125
 polling, 127
 pop and push, 124
 portable device programming, 425
 Portable Document Format (PDF),
 254–255
 portable operating system interface
 (POSIX), 421
 portable software, 416, 420–421,
 439–440
 Porter, Thomas, 30
 ports
 I/O, 97, 142–144
 IEEE 1284 parallel, 152
 RS-232 serial, 154
 positive logic, 5–6
 positive number representation, 6–8
 POSIX (portable operating system
 interface), 421, 439
 post function (jQuery), 345–346
 postfix notation, 125, 227
 PostScript language, 124, 254
 power consumption vs. performance, 138
 power series approximations, 313
 power wall, 119
 prefetching, 135
 prefix notation, 125
 prepress technologies, 29
 primitive data types
 arrays, 185–187
 bitmaps, 187–188
 overview, 184–185
 strings, 188–189
Principles of Compiler Design (Aho and
 Ullman), 228
 print servers, 337–338
 printers
 color system, 173
 and steganography, 363
 printf (print formatted) function
 (C), 277
 priority interrupts, 128
 privacy. *See also* personal data
 and data visibility, 378–379
 as security, 352
 privileged instructions, 133
 privileges, and security, 356
 PRNGs (pseudorandom number
 generators), 375
 procedures. *See* functions
 processes, 337–338
 processor cores, 119, 135
 processor interrupt handling, 341
 production grammars, 320–322
 program counter, 101–102
 programmable read-only memory
 (PROM), 85
 programmers. *See also* career success
 adding value, 414–416, 442, 443
 finishing projects, 419
 productive environment for,
 437–439
 training, 418, 426

programming. *See also* software hygiene discipline of, 428, 443
 Linux environment, 437
 productivity tools, 437–439

programming languages
 assembly language, 217–218
 compiler execution, 232–234
 compilers vs. interpreters, 228–229
 domain-specific, 228
 grammar, 226–227
 and hardware, 236
 high level, 219–220
 interpreter execution, 231–232
 and lexical analysis, 221–226
 optimizers, 234–235
 structured vs. unstructured, 220

Programming Pearls (Bentley), 228

programming projects
 documentation, 432
 fast prototyping, 432–433
 interface design, 433–436
 and library code, 436

programs. *See also* user programs
 development vs. maintenance, 349
 machine learning, 281–282
 running, 137–138
 testing, 440–441
 third-party code, 376–378
 and value proposition, 414–416

PROM (programmable read-only memory), 85

promise construct, 346–350

propagation delay, 43, 57, 70, 71

properties (C++), 211–212

prototyping, 432–433

proxies, 358

pseudo-instructions, 218

pseudocode, 122

pseudorandom number generators (PRNGs), 375

pseudorandomness, 318–319, 375–376

public key cryptography, 368

public key infrastructure (PKI), 370–371

punched paper tape, 84

push and pop, 124

Q

qsort function (UNIX), 213

quadrature encoding, 150–151

quadrees, 123, 304–310

quantization, 323–333

queues, 270, 272–273, 344

Quoted Printable (QP) encoding, 26

R

race conditions, 335–336

radial zones, 87

radians, 301

RAM (random-access memory), 82

ramp converters, 165

random back-off-and-retry, 158

random logic, 112

random number generators, 368, 375–376

randomness
 approximating, 318, 322–323
 dithering as, 325

raster graphics, 180–181, 311–312

raw buffer mode, 271

Raymond, Eric, 434

RCS (Revision Control System), 440

read-only memory (ROM), 83–85

real numbers, 14–18, 283

realloc function (C), 195–196, 379–381

recurrent neural network, 405

recursion, 122–125

recursive subdivision
 and constructive solid geometry, 304–311
 defined, 122
 drawing spirals with, 301–304

reduced instruction set computers (RISC), 113–114

Reed, Brian, 439–440

refactoring, 441

reference addressing, 101, 185, 198

reference voltages, 163–164

registers
 accumulator, 103–104
 in computer design, 133–134
 condition code, 10, 98
 index, 129
 indirect address, 110
 instruction, 109
 limit, 124
 memory, 79–81
 program counter, 101–102
 schematic, 78–79

regression testing, 441

regular expressions, 224–225

relative addressing, 129–130, 186

relays, 47–50

- reset bar, 72
 - Resig, John, 253
 - resistance (R), 44
 - resolution
 - CRT screen, 178
 - digital-analog conversion, 161, 165
 - graphics, 180, 291
 - reverse Polish notation, 125
 - Revision Control System (RCS), 440
 - RGB color model, 28
 - Riemersma dithering algorithm, 332–333
 - Riemersma, Thiadmer, 332
 - ring buffers, 272–273
 - ripple-carry adder, 62
 - ripple counters, 77
 - RISC (reduced instruction set computer) machines, 113–114
 - Ritchie, Dennis, 220, 288–289, 416
 - Rivest, Ronald, 368
 - ROM (read-only memory), 83–85
 - root (tree), 200, 243
 - rootkits, 354
 - Rosenblatt, Frank, 402
 - rotary encoders, 149–151
 - rotation mode, 316
 - rotational latency, 87
 - routers, 158
 - Rozin, Paul, 429
 - RSA algorithm, 368
 - Ruby language, 228
 - Rumelhart, David, 404
 - runtime libraries, 138, 274, 275–276
- S**
- S-R (set-reset) latches, 72–73
 - Samet, Hanan, 123
 - sample and hold circuit, 162–163
 - sampling
 - audio, 165–166, 168–170, 171
 - circuits for, 162–163
 - defined, 160
 - images, 173–174
 - SATA interface, 154
 - saturation regions, 41
 - Scalable Vector Graphics (SVG), 254–255
 - scaling (graphic), 291
 - scan lines, 311
 - scanners, 180
 - scatter/gather, 211
 - SCCS (Source Code Control System), 440
 - schematic diagrams, 44
 - scheme (URL), 240
 - Schmidt, Eric, 225
 - Schmitt, Otto H., 55
 - Schmitt triggers, 55, 56
 - Schwartz, Barry, 429
 - scientific notation, 15–16
 - <script> elements (HTML), 252
 - searching
 - databases, 205–206
 - with hash functions, 213–215
 - tree traversing, 199–203
 - security. *See also* cryptography; software hygiene
 - authentication and authorization, 361–362
 - of communications, 356–357
 - and internet, 357–359
 - metadata and surveillance, 359
 - objectives, 351–352
 - physical, 355–356
 - and society, 359–361
 - threat model, 352–353
 - trust violations, 353–355
 - seed, 375
 - selectors, 65–66
 - self-similarity, 319
 - Semi-Automatic Ground Environment (SAGE), 157
 - semiconductors, 51
 - sensors
 - digital camera, 38–39
 - rotating shaft, 149–151
 - sequential logic, defined, 69–70
 - sequential memory, 85
 - sequential shift register, 99
 - serial communications, 152–154
 - series connections, 43
 - set-reset (S-R) latches, 72–73
 - Sethi, Ravi, 228
 - setup time, 76
 - SHA-1 algorithm, 369–370
 - Shamir, Adi, 368
 - sharding, 216
 - shared libraries, 138
 - shared resources, 336, 337
 - shells, 437–438
 - shift operations, 99–100, 153
 - shift-reduce parsers, 226–227
 - shift registers, 99–100, 153

- shifting and masking, 311–312
- shortcuts *See* approximations and shortcuts
- sibling node, 243
- side-channel attacks, 378
- sigmoid neurons, 403–404
- sign and magnitude representation, 10–11
- SIGSALY voice encryption system, 367
- Simple Mail Transfer Protocol (SMTP), 159
- sine waves
 - approximation, 313
 - characteristics, 155
 - digital reconstruction, 165–166
- single-pole, double-throw (SPDT) switches, 45–46
- single-pole, single-throw (SPST) switches, 45, 48
- single-precision numbers, 17–18
- singly linked lists, 191–195
- slide rules, 34–35
- small-scale integration (SSI) parts, 60
- SMTP (Simple Mail Transfer Protocol), 159
- Snowden, Edward, 360, 368
- Sobel edge detection, 395–398
- Sobel, Irwin, 395
- SoC (system on a chip), 119
- social attack mechanisms, 358
- society and security, 359–361
- software distribution, 421–422
- software hygiene
 - and attack surfaces, 373–374
 - data as code, 382–384
 - error checking, 373
 - memory management, 374–375, 379–382
 - random number generation, 375–376
 - security measures, 372–373, 378–379
 - third-party code, 376–378
- solid-state disk drives, 88, 382
- sorting, 212–213
- source code control, 440
- Source Code Control System (SCCS), 440
- space-filling curves, 319–320
- spaghetti code, 220
- spam filters, 387, 389–390
- spatial data structures, 123
- spectrum analyzers, 168
- spinning, 341
- SQL implementations, 423
- SQL injection, 382–384
- square waves, 165–166
- SRAM (static RAM), 82
- stacks, 122–125
- Stallman, Richard, 418, 419
- standards proliferation, 238, 415
- Star Trek II: The Wrath of Khan* (film), 322–323
- state machines, 109, 112, 223–224
- state tables, 223, 224–226
- static data, 136
- static linking, 138
- static RAM (SRAM), 82, 95
- statistical analysis applications, 387, 410–411
- stderr file pointer, 274–275
- stdio (standard input/output) library, 271–272, 274
- stdlib library, 271–272, 274, 275
- steganography, 362–363
- Steinberg, Louis, 330
- stepper relays, 49
- stereo, 170–172
- stochastic processes, 322–323
- Stone, Maureen, 29
- storage technologies
 - block devices, 85–88
 - mixed devices, 88
 - read-only memory, 83–85
- storage tubes, 179
- stored procedures, 423
- strcmp function (C), 213
- strength reduction, 235
- string library, 275
- string terminators, 188–189
- strings
 - in C, 276–280
 - data structure, 188–189
 - sorting, 213
- strobe signals, 152
- Stroustrup, Bjarne, 212
- student projects, 419
- The Stuff of Thought* (Pinker), 432
- <style> element (HTML), 246–247
- subroutines. *See* functions
- substrates, 51
- subtractive color system, 28–29, 173
- successive approximation converters, 165
- Sun Microsystems, 420, 439
- supersampling, 174

- surveillance, 359
- SVG (Scalable Vector Graphics), 254–255
- switches
 - electrical, 44–47
 - networking, 157
 - in plumbing example, 43
- symbols, 2, 3, 221
- symmetric encryption, 364, 367–368
- synchronous counters, 77–78
- syntactic sugar, 189, 346, 347
- system calls, 133, 269–271, 343
- system integrators, 377
- system on a chip (SoC), 119
- system programming vs. application programming, 259, 282
- system space, 133

T

- table lookups
 - character classification, 288–290
 - conversion tables, 284
 - texture mapping, 285–288
- tags, 241–242, 248
- Talbot, Henry, 325
- tape technologies, 84, 85, 87
- Taylor series, 313
- TCP/IP (Transmission Control Protocol/Internet Protocol), 158–159
- Tektronix storage tubes, 179
- telephone networks, 157
- telephone technologies, 155
- teletype technology, 153–154, 176–177
- The Ten Commandments for C Programmers* (Spencer), 372
- terabytes, defined, 21
- terminal node, 243
- terminals
 - blit, 209
 - and buffering, 270–271
 - as coding interface, 437
 - hardcopy output, 176–177
 - screen based, 177–178
 - software implemented, 268–269
- test and set instruction, 342
- testing, 440–441
- Texas Instruments, 53
- text editors, 438–439
- text. *See* characters
- texture mapping, 285–288
- third-party code, 376–378, 436

- This Is Spinal Tap* (film), 39
- Thompson, Ken, 24, 224, 377, 416, 417, 434
- thrashing, 177
- threads, 338–339
- threat model, 352–353, 378
- thresholds
 - binary vs. decimal, 41
 - in graphics, 324
 - in hyperesis, 55
 - negative- and positive-going, 55–56
 - and transfer functions, 49
- Tiemann, Michael, 418
- time and date structure, 189–190
- time division multiplexing, 154, 157
- time references, 70–71
- time-sharing systems, 177
- timers, 128, 133
- timing attacks, 378
- tokens, 221, 225–226
- Torvalds, Linus, 418
- totem-pole outputs, 58
- touch devices, 181
- traffic control unit, 110–113
- transactions, 340–341
- transfer functions, 38–40, 49, 54–55, 168
- transformations (graphic), 291
- transistors, 51–52
- translations (graphic), 291
- Transmission Control Protocol/Internet Protocol (TCP/IP), 158–159, 211
- transparency
 - color, 29–30
 - open source code, 376–378
 - and security, 355
- transposition ciphers, 365–366
- trapdoor functions, 368
- tree balancing, 202–203
- tree lexicon, 243
- tree of knowledge, 260, 262
- tree traversal, 123, 244, 280–281
- trees. *See also* nodes; octrees; quadtrees
 - B-tree, 205
 - binary, 199–203
 - defined, 123
 - examples, 229–230, 242–243
- tri-state outputs, 60
- trigonometric functions, 301, 313–318
- triodes, 50–51
- troff (typesetting language), 228, 239, 274–275

- trust
 - and third-party code, 376–378
 - violations, 353–354, 361
- truth tables, 4
- Turing, Alan, 101
- twisted-pair cabling, 56
- two-factor authentication (2FA), 361–362, 373
- 2001: A Space Odyssey* (film), xxii, 386
- two's complement adder, 60–63
- two's complement representation, 13–14
- typeballs, 177
- typesetting languages, 228, 239, 274–275

U

- UART (Universal Asynchronous Receiver-Transmitter), 154
- Ullman, Jeffrey, 228
- underflow condition, 10
- Unicode standards, 24
- Unicode Transformation Format 8-bit (UTF-8), 24–25, 439
- Uniform Resource Locators (URLs), 27, 159, 239–240
- unions, 190
- Universal Asynchronous Receiver-Transmitter (UART), 154
- Universal Serial Bus (USB), 152, 156
- UNIX
 - API, 433–435
 - brief history, 415, 416–418
 - derivative operating systems, 437
 - interrupt mechanism, 128
 - sorting functions, 213
 - tools, 421
 - The UNIX Programming Environment* (Kernighan and Pike), 434
 - UNIX-to-UNIX copy (UUCP), 157
 - URL. *See* Uniform Resource Locator (URL)
 - USB (Universal Serial Bus), 152, 156
 - user interfaces (UI), 433
 - user programs. *See also* C programs; “Guess the animal” program
 - vs. browsers, 259–260, 273–274, 282
 - and operating system, 268–269
 - user space, 133, 337
 - UTF-8 (Unicode Transformation Format 8-bit), 24–25, 439
 - UUCP (UNIX-to-UNIX copy), 157

V

- vacuum tubes, 50–51
- van Eck phreaking, 378
- Vannini, Walter, 428
- variables
 - FORTRAN naming conventions, 219
 - local, 124
 - and size assumptions, 374
- vector graphics, 178–181
- vectored I/O, 210–211
- vectoring mode, 316
- video, 174–176
- virtual machines, 229, 237, 425
- virtual memory, 132
- voice encryption, 367
- volatile keyword (C), 236
- Volder, Jack, 313
- Volta, Alessandro, 44
- voltage (V), 43–44
- von Neumann architecture, 118
- von Neumann, John, 118
- voxels, 310

W

- Wall of Sound concert audio system, 57
- WAN (wide area networks), 156–157
- WarGames* (film), 383–384
- Warnock, John, 254
- wave characteristics, 154–157
- waveform generation, 162
- web browsers
 - overview, 256–257
 - vs. programs, 259–260, 273–274, 282
 - and standards, 238
 - as virtual machines, 237
 - and World Wide Web, 159
- web pages. *See also* Cascading Style Sheets (CSS)
 - advertising pixels in, 363
 - asynchronous issues, 344
 - canvas, 290–291
 - comment sanitization, 384
 - Document Object Model, 242–244
 - “Guess the animal” game, 263–267
 - HTML documents, 240–242
 - and markup languages, 238–239
 - overview, 159–160
 - styling, 244–248, 267
- Weinberger, Peter, 438
- whitespace, 228

Wi-Fi, 158
wide area networks (WAN), 156–157
Williams, Lance, 285–286
Williams, Roland, 404
wired-AND gates, 59
Wizard of Oz, The (film), xxiv
word, defined, 21
World War II code breaking, 357,
 366–367, 378
World Wide Web, 159
writable control store, 113
Wrzesniewski, Amy, 429

X

X Window System, 341, 439–440
x-y coordinates. *See* Cartesian
 coordinate mapping
Xerox Palo Alto Research Center
 (PARC), 158

XHR (XMLHttpRequest), 344
XHTML, 239, 242
XML Path Language (XPath), 250
XML. *See* eXtensible Markup Language
 (XML)
XMLHttpRequest (XHR), 344
XOR (exclusive-OR)
 logic gates, 53
 operation, 4–5, 9
Xtensible Stylesheet Language
 Transformations (XSLT),
 250–251

Y

yacc program, 226–227, 229–230

Z

Zim, Herbert, 364