Index

'The Hub', 19
15c3-5, 7
1977 trinity, 80, 115
3D graphics, 269
60 order code, 139

a posteriori knowledge, 68
a priori knowledge, 68
A-0 system, 34
Aarhus, 177
Aaron, Sam, 124
ABM, *see* anti-ballistic missile system; anti-ballistic missile defence
abstract data type, 193, 211, 212, 215, 230, 237, 257, 263, 299
abstract idea, 60
abstract machine, 207
abstraction, 236, 288, 291–294, 300, 304
academia, 40
academic computer scientist, 195
academic paper, 71
academic publication, 44, 286, 293
acceptance condition, 155
acceptance testing, 149, 158, 165, 183, 186
accidental complexity, 23
accidental tasks, 173
accountability, 28
accounting, 25
ACL2, 6, 65
ACM, 10, 40, 44, 156, 199, 201, 268
ACM Symposium on Principles of Programming Languages, 62, 153
activity, 243
activity declaration, 243, 256
Actor model, 95
actor-oriented approach, 273
Ada package, 275
Ada, Countess of Lovelace, 88, 275
adaptation, 182
Advanced Research Projects Agency, 89, 128, 250, 278
advising, 94
aerodynamic stability, 83
aesthetics, 86, 252, 294
agent-based simulation, 78
Agile, 2
Agile development, 238, 272
Agile development methods, 163
Agile manifesto, 2, 134, 238, 277
Agile methodologies, 124, 134, 277
Agile Software Development, 18
Agile software development, 288
AH-6 helicopter, 11
AI languages, 179
air defence, 13, 84
Air Force, 145
air-traffic control, 174
aircraft, 67
Airy integral, 141
Alexander, Christopher, 182
ALF theorem prover, 222
algebraic effect handlers, 154
algebraic language, 118
Algol, 3, 11, 44, 71
  criticism, 34
  formalisation, 34
  preliminary report, 33
Algol 60 report, 291
Algol 68, 16, 54
Algol 68 Minority Report, 148, 155, 290
Algol report, 45, 94
Algol working group, 54
Algol, compiler, 48
Algorave, 3
algorithm, 4, 33, 40, 72, 125, 166, 174, 248, 293, 305
  in the Communications of the ACM, 47
  trading, 5
algorithmic bias, 4, 24, 305
Algorithmic Language, *see* Algol
algorithms, 26
alienation, 24
Allen, Paul, 115
almost a new thing, 236
Altair 8800, 80, 113, 115, 122
alter tape, 145
Amazon, 4
American Airlines, 168

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

American management philosophy, 59
American school, 260, 263, 264
amplifying human intelligence, 213
An Open Letter to Hobbyists, 115
analog computer, 12, 83, 88, 137
analog system, 170
Andersen, Arthur, 174
anecdotal evidence, 168
animation system, 107
anthropomorphic metaphor, 42
anthropomorphisation, 99
anti-ballistic missile, 171
anti-ballistic missile defence, 166–303
anti-ballistic missile software, 176
anti-ballistic missile system, 135
anti-intellectualism, 59
anything goes, 102, 306
Apollo, 2, 15, 21
Apple, 268
Apple Computer, Inc., 115
Apple II, 115, 119
applet, 271
applicability, 289, 290
application development lifecycle, 275
application software, 160
applicative expressions, 207
applied mathematics, 1
apprenticeship, 39
architecture, 286
Armstrong, Joe, 185
ARPA, *see* Advanced Research Projects Agency
ARPA community, 103, 126, 247
ARPANET, 3, 12, 103, 240, 297
array, 199
art, 122–125, 293
Art and Illusion, 110
art collectors, 97
articulate language, 110
artificial intelligence, 45, 65, 86, 87, 94, 97, 213, 256
Artificial Intelligence System, 269
artist, 18, 87
As We May Think, 18, 88, 100, 294
Aspect-Oriented Programming, 94
assembly, 58
assembly language, 67, 145, 258
assertion, 264
Association for Computing Machinery, *see* ACM
Association of Simula Users, 266
associative trail, 88
assumption, 287, 292, 304
Atkinson, Bill, 121
atomic symbol, 249
Auerbach, Al, 140
Augmentation Research Centre, 101
augmenting human intellect, 78, 87–91, 100–102, 121, 127, 247, 297
augmenting the human mind, 78
authentication problem, 210
auto-complete, 194
Autocode, 34
autognostic objects, 257
automated test runner, 183
automated testing tools, 133
Automath theorem prover, 220, 222
automatic coding, 34, 41
autonomy, 179
axiom, 52, 63
Axiom of Assignment, 52
axiomatic approach, 49
Axiomatic Basis for Computer Programming, 48

backdoor, 116
Backus, John, 16, 38, 198
Backus–Naur form, 50, 203
Ballistic Missile Early-Warning System, 67
ballistics tables, 137
bandaids and tourniquets, 167
bankloan, 245
Bartik, Jean, 36, 137
BASIC, 80, 115–118
basic assumption, 9, 77, 304
BASIC manual, 116
batch processing, 11, 19, 83, 91, 144, 214
Bauer, Friedrich L., 148
BBN, *see* Bolt Beranek and Newman
BBN-LISP, 93, 97, 147
BDFL, 186
beauty, 294
Beck, Kent, 182, 238, 269, 273
Beginner's All-purpose Symbolic Instruction Code, 115
Belady, Les, 174
belief, 290, 292, 298
Bell Labs, 12, 174
bells and whistles, 167
Bemer, Bob, 200
Benevolent Dictator for Life, 186
Berkeley, 100
Bertrand, Russell, 191
better old thing, 257, 267
bias, 24, 287
big technology companies, 79
Biggs, E. Power, 240
binding, 249
Birkbeck College, 206
bit fiddling, 192
bit representation, 295
black art, 16, 43, 55, 115, 141, 147, 148, 156, 157, 177, 181, 183, 184, 186, 198, 199, 277, 301
black art of programming, 16, 132
black box, 157
blackboard, 71
blog post, 7

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

blue book, 268
Blue Marble team, 121
BNF, *see* Backus–Naur form
Bobrow, Daniel G., 97, 256
Boehm, Barry, 176
Bolt Beranek and Newman, 88, 92, 97, 145
Booch, Grady, 274–277
Boolean logic, 214
bootcamp, 26
bootloader, 113
bootstrapping, 240
Borning, Alan, 262
boundary object, 44, 60, 184, 185, 286
Boxer, 109–112, 122
Boxer project, 261
Boyer, Robert S., 65
Boyer–Moore theorem prover, 65, 67, 214
branching point, 157
Brand, Stewart, 104, 114, 240, 250
Bravo editor, 119
break command, 144
breakpoint, 132, 144, 184
Bricklin, Dan, 119
Bright, Herbert, 168
British school, 260
Brooks' law, 134, 172
Brooks, Fred Jr., 134, 159, 171, 175
Brown, Henry, 192
bug, 133, 143
Building 26, 81, 85
Bush, Vannevar, 12, 18, 88, 100, 294
business data processing, 152, 192, 199–202
business journals, 294
Business Object Design and Implementation, 179
business record, 200
business tool, 119
business-oriented programming, 202
business-oriented programming language, 200
Byte magazine, 259
Böhm and Jacopini, 58
Böhm–Jacopini theorem, 35

C with Classes, 265
CAD system, 101
Calculus of Constructions, 223
Cambridge, 141, 213
Cambridge University, 265
Cannon, Howard, 278
Canon camera, 178
CAR, 93
Cardelli, Luca, 216, 260
Carnegie Mellon University, 224
Cartesian coordinates, 98
Cartesian product, 203
catch function, 154
category theory, 230
cathode-ray tube, 81, 83
causal effects, 11
CDR, 93
Cedar programming environment, 209
Certificate in Data Processing, 40
certification, 49, 61
Chalmers University, 223, 224
Chang, Hasok, 289–290, 298
change rate, 169, 186
changing requirements, 169, 182
character set, 267
chatbots, 1
ChatGPT, 28
checking, 143
checking mechanism, 299
checkout, 150
chemical revolution, 298
chess, 39, 174
chief programmer, 59, 64
Chief Programmer Team, 58–60, 62, 63, 161
children, 19, 97–100, 246, 255
chip, 69
Chrysler Comprehensive Compensation System, 182
Church, Alonzo, 191, 196, 203, 206
Church–Rosser theorem, 208
cigarette package, 247
City University, 213
Clark, Wes, 90
class, 236, 238, 243, 252, 262, 275
class browser, 261, 270
Class View, 227
Class-Responsibility-Collaboration (CRC), 274
Cleanroom, 161
Cleanroom methodology, 70, 218
Cleanroom Software Development, 63–66
clever code, 295
cluster, 193, 211
co-opting, 300
COBOL, 44
CODASYL, 200
code, 285, 292, 295
code block, 243
code completion, 227
code coverage, 186
code inspection, 158
code listing, 80
code of ethics, 25, 59
code poem, 295, 296
code walkthrough, 158
coding education, 289
coding vs. programming, 41, 137
cogency, 289, 290
cognitive capacity, 293
Cohn, Avra, 69
collaboration, 293, 300
colorful jargon, 167

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

combinatory logic, 209
COMEFROM statement, 59
command and control, 89, 102, 289
commercial art world, 97
commercial interest, 79, 127
commercial tool, 118
Committee on Data Systems Languages, 200
Commodore PET, 115
Common Business-Oriented Language, *see* programming language, COBOL
Common business-oriented language, *see* programming language, COBOL
Common Lisp Object System (CLOS), 279
common sense, 182
common wisdom, 168
CommonLoops, 278
communication, 100, 159, 177, 251
communication overhead, 134, 172
communication satellite, 248
communication system, 89
Communications of the ACM, 10, 40, 47, 49, 62, 68, 69, 153, 169, 207, 242
Compatible Time-Sharing System, 92, 146
CompCert, 34
CompCert compiler, 71, 225
compilation trick, 298
compiler, 41
compiler-compiler, 258
complementary science, 304
complex computer systems, 159
complex phenomena, 288
complex software, 72
complex software system, 186
complex systems, 167
complexity, 168, 238
  accidental, 23
  of computer programs, 62
component, 237, 273
component subprogram, 162
component test, 149
Composition with Lines, 96
compositional reasoning, 72
compositionality, 48, 57
computational media, 294
computer, 67, 250, 293
  ACE, 141
  Altair 8800, 80, 115
  analog, 83
  Apple II, 115
  BINAC, 140
  Burroughs 220, 247
  business applications, 13
  Commodore PET, 115
  Dynabook, 236
  EDSAC, 132, 140, 147
  ENIAC, 36, 137
  Feranti Orion, 207
  ground control, 29
  IBM 701, 199
  IBM 704, 91, 197
  IBM 7044, 145
  IBM 709, 91
  IBM 7094, 146
  installation managers, 43
  LINC, 90, 248
  Magnolia, 268
  military applications, 13
  MONIAC, 137
  onboard guidance, 21, 29
  operators, 37
  PDP-1, 12, 81–83, 91, 97, 132, 144
  Radio Shack TRS-80, 115
  tablet, 79
  TX-0, 12, 77, 81–86, 126, 132, 144
  TX-2, 89
  UNIVAC I, 41, 199
  user groups, 43
  Whirlwind, 12, 77, 83–86, 88
  Xerox Alto, 253
computer art, 20, 95–97
computer bug, 136
computer culture, 264
computer games, 118
computer industry, 62, 148
computer installation, 132
computer memory, 213
computer music, 3, 85
computer on a chip, 113
computer operator, 81, 137
computer people, 303
Computer Professionals against ABM, 168
Computer Professionals for Social Responsibility, 67
computer program, 285, 292
Computer Program Test Methods Symposium, 156
computer programming sector, 39
computer revolution, 6
computer science, 25, 30, 231, 289, 304
  curricula, 25
computer scientist, 295
computer scientists, 40
computer simulation, 241
computer, TX-2, 19
computer-related accidents, 136
Computers and Automation magazine, 147, 160
Computers and Democracy, 177
computing centre, 43
computing folklore, 9
computing industry, 159, 259
COMTRAN system, 199–201
concept stretching, 193, 218
conceptual flaw, 227
conditional, 250

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

conditional expression, 47
conference, 292
conference talk, 286
conjunction, 221
considered harmful, 297
constructive mathematics, 242
constructive proof, 223
Content Analysis Language, 102
context, 286
continuous testing, 186
contract, 237
control, 287
control structure, 273
control transfer, 56
controversial debate, 72
Cook, William R., 257, 260
Coq theorem prover, 224
Coquand, Thierry, 223
CORBA, 271
core assumption, 70, 72
core dump, 142
correctness, 150, 214
correctness guarantee, 194
correctness proof, 185
counter-culture, 12, 78, 104
counter-example, 51, 212, 230
counterfactual fairness, 4
CPAABM, 168
CPT, *see* Chief Programmer Team
craft, 72
CRC card, 274
creative freedom, 3
creative thinking, 194
creative thought, 110
creativity, 88, 98, 110
critical reflection, 96, 287
cross-machine compatibility, 43
CRT, 83, *see also* cathode-ray tube
cryptic reference, 294
CTSS, 92, *see also* Compatible Time-Sharing System
cultural marker, 292, 293
cultural pointer, 111, 294–298
culture, 292
culture of programming, 8
  interaction, 8, 9
  outputs, 17
cultures
  interactions between, 60
  meeting, 60
cultures of programming, 212, 228, 289, 303–305
  as an abstraction, 27
  bridging, 209
  clash, 118, 175, 239, 277, 287, 300–303
  interaction, 27–28, 66, 72, 95, 125, 178, 185, 193, 212, 217, 228, 230, 255, 259, 267, 277, 287, 300–303
  intersection, 87
cultures of proving, 63–66
Cunningham, Ward, 182, 238, 268, 273
Curry–Howard correspondence, 221, 222
custom operator, 258
customer, 22, 133, 182, 186, 194
customer involvement, 182
customisable syntax, 258
cybernetics, 25, 29, 42
cyberspace, 271
cyclomatic complexity, 157

Dahl, Ole-Johan, 203, 241–246
Damn Warren's Infernal Machine (DWIM), 95
Darley, D Lucille, 145
Dartmouth, 86, 91, 115
data abstraction, 260, 263, 267
data analysis, 81
data description, 200
data division, 200
data privacy, 24
data processing, 40, 201, 289
data processing industry, 39
data space, 192, 203, 212
data structure, 192, 275
data type, 136, 191, 213, 217, 291
database, 119
Datamation, 40, 59, 160
dBase, 120
DDT utility, 132, 144
de Bruijn, Nicolaas Govert, 220
de-orbit burn, 29
dead moth, 143
deadlock, algorithm correctness, 47
DEBUG tool, 145
debugger, 299
debugging, 46, 132, 133, 136–151, 156, 184–185, 268, 286, 287, 299
debugging mode, 148
DEC, 82, 268
DEC Debugging Tape, 144
decentralisation, 114
decision making, 182
decision theorist, 107
deductive argument, 69
deductive methods, 34
defensive programming, 155
degree, computer science, 25
DeMillo, Lipton and Perlis, 35, 61, 64, 68, 70, 72, 167, 218
DeMillo, Richard, 61
demonstration-oriented period, 163
Dennis, Jack, 84, 144
Department of Defense, 89
dependent function, 221, 222
dependent pair, 221, 222
dependent type, 193, 224

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

dependently typed programming languages, 225
deployment, 6, 175
deployment automation, 7
derivation rule, 63
design by contract, 264
design committee, 266
design requirement, 162
desktop metaphor, 79
destruction-oriented period, 157, 183
Deutsch, L. Peter, 92, 104, 269
developer tooling, 226–229
development
  iterative, 1
  top-down, 35
development browsers, 270
development methodology, 1, 185, 263
development process, 15, 55, 133, 149, 158, 175, 270, 301
DevOps, 6
diagnostic log, 144
differential analyser, 12, 88
differential equation, 12, 88
digital electronic computers, 137
digital system, 170
Digitalk, 269
Dijkstra, Edsger, 17, 22, 33, 40, 55, 57, 58, 60, 80, 118, 133, 150, 175, 186, 203, 277, 288–290, 300
disagreement, 287, 290
disciplinary repertoire, 166–175, 303
discipline, 298
discrete simulation, 245
discriminated union, 204
diSessa, Andrea, 111
disjunction, 221
division of labour, 14
Do What I Mean (DWIM), 94
do-it-yourself, 114
doctrine of types, 196
documentation, 149, 172, 194, 226
DoD, *see* US Department of Defense
DOD-STD-2167A, 165
domain failure, 154
domain-specific language, 102, 215
double meaning, 295
Draper Prize, 266
Dynabook, 78, 105, 106, 236, 247, 254, 257
dynamic lookup, 236
dynamic memory allocation, 244
dynamic typing, 236

E (editor), 180
E-programs, 174, 178
East coast, 100, 180
Eastern annual Joint Computer Conferences, 201
Eckert, J. Presper, 36–39
Eckert–Mauchly Computer Corporation, 140, 199
eclecticism, 231
Eclipse development environment, 272
economy, 137
ECOOP conference, 259
Edinburgh, 65, 214
Edinburgh LCF, 214
EDSAC, 34, 38, 56
education, 95, 97, 250, 288, 293
  liberal arts, 4
educational materials, 122
educator, 18, 87
Edwards, Dan, 83
effect handlers, 154
effect system, 219
efficiency, 219, 226, 295
efficient compilation, 227
Eiffel tower, 264
EJCC, 201
Electric Pencil, 119
electrical engineering, 14, 25, 289, 291
electronic brain, 42
electronic circuit, 107, 110
electronic computer system, 212
electronic music, 122–125
elegance, 294
ELIZA, 28
EMACS, 180
Emacs editor, 224, 226
empirical evidence, 168
end-user application, 120
Engelbart, Douglas, 77, 78, 100, 121, 247, 250, 294
engineering, 55, 276, 289
  traditional, 16
engineering culture, 16–18, 22, 25, 41, 54, 57, 67, 72, 139, 156, 166, 167, 174, 205, 210, 212, 217, 219, 226, 227, 230, 260, 263, 271, 274, 293, 295
engineering discipline, 16, 293
engineering enterprise, 264
engineering interest, 127
engineering knowledge, 286
engineering laboratory, 209
engineering perspective, 62
engineering practices, 16, 136
engineering reports, 156
engineering trade-off, 228
engineers, 127
English language, 200, 238, 274, 285
English, Bill, 102
English-like notation, 203
ENIAC, 10, 36
  women, 37
ENIAC programmers, 36–39, 137
ENIAC reprogramming, 139
EnS, 56
ensure clause, 265

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

enterprise software development, 273
entity, 298
environment, 168, 169, 174, 181
  forgiveness of, 23
environment division, 201
epistemic activity, 298
epistemological anarchism, 230
epistemological theory, 231
epistemology, 252
epoch of debugging, 187
Ericsson, 185
error checking, 227
error guessing, 157
error handling, 147
error message, 100
error rate, 36
error-free programs, 21
errorset function, 147, 151, 154
essential tasks, 173
Ethernet, 112
EVAL mode, 180
evaluation strategy, 219, 250
evaluation-oriented period, 165
Evans, Dave, 248
Evans, Thomas, 145
evolution of a meaning, 137
evolving environment, 186
exception, 153, 191, 264
exception handler, 147
exception handling, 151–154, 185, 264
executable specification, 183
exhaustive testing, 156, 170
existential quantification, 221
Expensive Planetarium, 91
Expensive Synthesiser, 91
Expensive Typewriter, 91
experiment, 133
exploration, 126
extensible language, 205
Extensible Languages Symposium, 282
extensible programming language, 205, 250, 258, 267
extension, 220
external problem, 293
Extreme Programming, 17, 134, 181–184, 238, 269, 272

face-to-face interaction, 186
factorial, 48, 57, 63, 111
factorial function, 46
fairness, 4
fault detection, 158
fault tolerance, 154–156
fault-tolerant programs, 153
fault-tolerant systems, 155, 185
Feranti Orion, 207
Ferranti, 174
Fetzer, James, 68
Feurzeig, Wallace, 97
Feyerabend, Paul, 105
Fifty Years' War, 277
file, 200, 205
file system, 219
filing cabinet, 192, 212
financial market, 6
financial regulations, 6
financial software, 6, 119
FIND program, 48
firing tables, 37
First Draft of a Report on EDVAC, 141
Fisher, David, 258
fixed point combinator, 209
fixed point mode, 191
fixed point type, 197
Flavors, 278
FLEX machine, 246
FLEX system, 246–251
flexibility, 237, 256
Flexowriter Interrogation Tape, 144
Flexowriter terminal, 81
flight control, 67
flight simulator, 83
FLIP, 94
Flit debugger, 84
FLIT utility, 132, 144
floating point mode, 191
floating point number, 295
floating point type, 197
floor turtle, 98
flowchart, 84
FLOW-MATIC, 42
FLOW-MATIC system, 199–201
flowchart, 158, 248
Floyd, Christiane, 134, 177, 181, 261
Floyd, Robert W., 47, 62
folk history, 246, 257
folklore, 12
foo, bar, baz, 296
foreign country, 285
forgiving environment, 168
formal definition, 51
formal elegance, 261
formal grammar, 44
formal language, 1, 8, 42, 126
Formal Language Description Languages, 207
formal logic, 191, 196–198, 208
formal methods, 6, 135–137, 218, 286, 301
formal proof, 70, 168, 194
formal reasoning, 166
formal specification, 35, 186, 305
formal system, 195, 196
formal verification, 63, 67, 69, 133, 302
  of financial algorithms, 6

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

format string, 224
Forrester, Jay, 83
FORTRAN, 10, 16
foundations of mathematics, 195–197, 222
Foyles bookshop, 206
Frankston, Bob, 119
free society, 23
free software, 13, 114
Free Software movement, 86
freedom, 301, 305
  software, 24
Frege, Gottlob, 196
functional language, 191, 195, 216
functional programmer, 218
functional programming, 214, 223, 273
fundamental knowledge, 33
fundamental law, 166
fundamental research, 201
funding
  US government, 63
future of programming, 28

Gabriel, Richard P., 94, 180
Galison, Peter, 291
Gamma, Erich, 182
garbage collection, 265
garbage collector, 244
Garmisch, 148
Garmisch, Germany, 55
Gates, Bill, 115
GDPR, 4
Gelperin, David, 156
gender bias, 33
GHC compiler, 230
giant brain (nickname), 36
Gill, Stanley, 142
Glennie, Alick, 34
glitch, 6
Glitch art, 21
GNU project, 13
Go sub statement, 116
Go to considered harmful, 17, 35
Go to statement, 116
Go to statement considered harmful, 57
go wrong, 193, 215, 217, 226
Goldberg, Adele, 19, 20, 106, 253, 254, 267
golden event, 291
Goldstine, Adele, 137
Gombrich, Ernst, 110
Goodenough, John B., 153–157
Gosling, James, 271
GOTO, 35
goto, 53
goto statement, 56, 57
gradual development, 170
Graetz, Martin, 81
GRAIL system, 90, 248
grant application, 71
graphical display, 247
graphical interface, 108
graphical user interface, 100, 110, 248, 261, 271, 301
Greek letter, 297
Green project, 271
Greenblatt, Richard, 94
Guidance and Navigation System Operations Plan, 15
guideline, 294

H-LAM/T, 101
hacker, 3, 12, 77, 295, 301
  microcomputer, 126
  MIT, 126
hacker community, 97
hacker culture, 11–13, 24, 41, 57, 72, 83, 93, 145, 148, 179–182, 198, 220, 226, 230, 246, 252, 265, 266, 277, 286, 289, 291, 293, 295, 297, 300
hacker ethic, 83
hacker practice, 181, 185, 299
HAKMEM, 12, 86, 105, 252, 294
HAKMEM memo, 297
Halpern, Mark, 147, 184
halting problem, 166, 186
hardware, 126, 250, 302
hardware architecture, 272
hardware device, 66
hardware errors, 38
hardware failure, 140, 185
hardware fault, 21
hardware verification, 216
Harvard, 172
Harvard Business Review, 178
Harvard Mark II, 143
heavyweight process, 238
Heidegger, 4
Heideggerian hammer, 29
Hello world, 297
Hetzel, Bill, 156
heuristic, 65, 86
Hewitt, Carl, 95
Hewlett-Packard, 268
hierarchical structure, 59
hierarchy of types, 196
high modernity, 24, 187
high school, 107
high-integrity system, 237, 264
high-level programming language, 172, 174, 184, 241, 258
higher-kinded quantification, 220
Hindley–Milner algorithm, 210, 215
Hingham Institute, 81
historical actor, 290
historical period, 285

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

history of programming, 288–290
history of science, 304
history of types, 229
Hoare triple, 48, 56, 65, 265
Hoare, C. A. R., 47, 52, 60, 70, 135, 156, 203, 242, 244, 288–289
hobbyist, 123, 289
hockey, 271
Homebrew Computer Club, 112–115
homoiconicity, 146
Hopper, Grace, 34, 42, 143, 199
hospital simulation, 107
hostile environment, 168
Huang, J. C., 157
Huet, Gerard, 223
human augmentation, 289
human computers, 10
human factors, 89
human–computer interaction, 136, 177
human-readable text, 296
humanistic culture, 18–20, 23, 24, 87, 100, 105, 107, 120, 178, 180, 246, 250–254, 277, 293, 295, 297, 301, 305
hydraulic devices, 137
hydrogen bomb, 37
HyperCard, 120
hypertext, 88, 294

i, 47
I/O, 54
IAL, 44, *see also* International Algorithmic Language
IBM, 35, 53, 59, 63, 86, 118, 148, 152, 155, 161, 171, 174, 200
IBM 704, 42, 87
IBM OS/360, 159
IBM Research, 278
M Visual Age, 270
IBM System/360, 152
IBM Vienna lab, 153
IBM VisualAge, 272
IBM Visual Age Smalltalk, 270
icon, 110
IDE, *see* Integrated Development Environment
idiosyncratic typography, 205
IEEE, 165
IEEE Computer, 173
If you See What I Mean, 208
IFIP WG 2.3, 54
IFIP Working Group 2.1, 204
image tradition, 291
immediate feedback, 225
imperative programming, 274
Imperial College, 174
implacable sceptic, 65
implementation, 244
incommensurability, 23, 27, 87, 236, 285, 291, 298, 302
Incompatible Timesharing System, 146
incremental compilation, 133
incremental models, 176
indirect addressing, 141
Indo-European languages, 248
industrial factories, 161
industrial production line, 161
industrial research, 54
industrial-grade component, 264, 265
industry, 62
industry crisis, 39, 148, 259, 263
industry practice, 132
industry standard, 165
Infer Static Analyser, 65
inference rule, 193, 215
informatics, 25
information exchange, 298
information hiding, 136, 153, 169, 193, 210, 273, 299, 300
Information is free, 181
information processing management, 160
Information Processing Techniques Office, 89, 102, 128
information science, 25
information systems, 25
Ingalls, Dan, 108, 253, 254
ingenious tricks, 61
inheritance, 236, 237, 252, 255, 256, 260, 263
inheriting, 245
initial orders, 142
Institute of Electrical and Electronics Engineers, 165
instruction-by-instruction, 142
instruction-by-instruction debugging, 132
integer type, 197
integral equations, 60
integrated computational environment, 112
Integrated Development Environment, 78
integration testing, 15, 165, 183
Intel, 113, 268
Intel 4004, 113
Intel 8080, 113
Intel iAPX 432, 259
interaction
  collaboration, 18
  struggle for control, 18
interactive computing, 132
interactive editor, 225
interactive essay, 294
interactive games, 122
interactive programming, 77–81, 174, 179, 225, 291, 300, 301, 303
interactive programming environments, 182
Interactive Software Engineering, 264
interactive terminal, 116

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

interactive theorem proving, 127, 220–226
interdisciplinary, 293
Intergalactic Computer Network, 240
interim Dynabook, 106
interim language, 200
Interlisp, 94, 109, 117
internal memo, 294
International Algorithmic Language, 199, *see also* IAL
international standard, 286
International Symposium of Formal Methods, 135
Internet, 12, 103, 134, 240, 272, 297
interpreter, 41, 51, 146
interpretive checking routine, 142
interpretive routines, 34
intuitionistic logic, 195, 221
invalid program, 210
inverse square root, 295
iPad, 106
IPTO, *see* Information Processing Techniques Office
irrelevant detail, 289
ISO C++ standard, 266
ITS, *see* Incompatible Timesharing System

Jacobson, Ivar, 276
jargon file, 297
Jennings, Betty Jean, *see* Bartik, Jean
job interview, 26
Jobs, Steve, 115
Joint Computer Conference, 100
journal, 292
Joy, Bill, 271
juggling, 100
JUnit framework, 272
Jupyter notebook, 81

Kay, Alan, 19, 78, 105–109, 239–241, 246–250, 254, 278, 300, 301
Kemeney, John G., 115
Kennedy, Andrew, 229
kernel, 67
kilogram, 229
KIM-1 microcomputer, 122
kitchen sink, 167
Knight Capital, 1, 5, 8, 21, 23, 65
Knight, Thomas, 94
knowledge, 39–41, 83, 179, 286, 292–294, 298–300
  practical human, 4
knowledge exchange, 299
Knowledge Representation Language, 256
Knuth, Donald, 40, 220, 242
Kotok, Alan, 81, 144
Kuhn, Thomas, 217, 290
Kurtz, Thomas E., 115

labour crisis, 14, 33, 34
Lakatos, Imre, 61, 212
lambda calculus, 34, 45, 52, 87, 191, 193, 195, 203, 206, 221, 222, 250, 260
Lambda Days Conference, 195
lambda notation, 93
Lambda the Ultimate, 95
Lamport, Leslie, 62
Lampson, Butler, 119
Landin, Peter, 206, 250
language metaphor, 42
language safety, 226
language specification, 293
large corporations, 59
large language models, 28
large software systems, 156, 273
large-scale programming, 136
laser printer, 112
Latour, Bruno, 267
laws of physics, 60
lazy evaluation, 219
LCF, 65, 214
LCF theorem prover, 193, 220
learning, 126, 177, 179, 181, 293
legacy system, 44
legal constraint, 287
legibility, 165
Lehman, Meir, 174
Lensman series, 81
letters to the editor, 62
levels of language, 298
lexical scoping, 95
liberal arts, 4
Library of Congress, 82
library problem, 88
Licklider, J. C. R., 88, 101, 103, 121, 168
lifecycle, 1, 275, 276
light pen, 19, 81, 83
lightweight formal method, 193, 218
lightweight specification, 133, 225
limits of software, 185
line numbers, 117
linear equation, 116
linguistics, 42
Linux, 14
Lipton, Richard, 61, 158
Liskov, Barbara, 153, 211, 256, 263, 288
LISP, 45, 65
Lisp, 92–95
LISP 1.5, 151, 154
LISP editor, 92–95, 126, 146, 180
Lisp machine, 95, 179, 278
Lisp system, 184
Lisp wizard, 180
Lisp-Flexo system, 92
list processing, 45, 86, 92, 98
literacy, 1, 18, 293
live coder, 125

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

live coding, 3, 21, 80, 122–125
live editing, 238
Lively Kernel, 239
Lockheed, 148
logging, 142
logic, 66, 213, 291
Logic of Computable Functions, 214
Logic Theorist, 65, 86
logic tradition, 291
logical argument, 169
logical paradox, 193
logical proposition, 221
logical tradition, 193
logical type, 213, 217
Logo, 19
love, 100
Lovelace, Ada, *see* Ada, Countess of Lovelace
LSD, 78, 104, 121
LSD trip, 104
Lucid Energize, 227

machine, 288
machine checking, 72
machine independence, 8, 43
machine instruction, 57
machine language, 42, 258
machine learning, 305
Machine-Oriented Language, 102
Macintosh, 120
MacKenzie, Donald, 136
macro, 259
Macro assembler, 84
magazine, 80, 294
magnetic core memory, 84
Magnolia, 268
mailing list, 292
mainframe computer, 118
malleability, 263
man–computer symbiosis, 78, 88, 94, 194
man–machine symbiosis, 121, 127
management, 59
management process, 132
management structure, 64
manager, 303
managerial concept, 193
managerial culture, 13–15, 22, 25, 41, 56, 105, 114, 143, 148, 159–166, 179, 212, 275, 278, 285, 291, 292, 300
managerial interest, 127
managerial method, 62, 273
managerial reports, 156
managers, 13, 59, 127
Managing the Development of Large Software Systems, 163
manufacturing, 241
many things go, 306
map function, 219
mapping, 180
Mariner 1 spacecraft, 159
Martin-Löf, Per, 222
masculinisation
  of software engineering, 29
master programmer (ENIAC), 38
mathematical
  entity, 1
  thinking, 1
mathematical abstraction, 67
mathematical analysis, 170
mathematical culture, 10–11, 21, 41, 57, 62, 63, 67, 87, 105, 118, 136, 148, 151, 152, 155, 156, 195, 203, 205, 206, 209, 212, 217–219, 222, 225, 226, 228, 242, 260, 285, 291, 292, 296, 300, 302, 305
  characteristics, 10
  subcultures, 70
mathematical entity, 68, 285, 292
mathematical idea, 296
mathematical law, 169
mathematical logic, 206
mathematical methods, 45, 71, 72, 156, 290
mathematical model, 68, 222
mathematical notation, 285, 297
mathematical object, 192
mathematical problem, 287
mathematical proof, 61, 66, 67, 169, 302
mathematical puzzle solvers, 39
mathematical reasoning, 136
mathematical relation, 229
mathematical rigour, 66
mathematical software, 269
mathematical structure, 202
mathematical theorem, 166
mathematical verification, 64
mathematics, 98, 289
Mauchly, John, 36–39
McCabe, Thomas J., 157
McCarthy, John, 45, 46, 51, 64, 71, 86, 179, 191, 202, 213, 290
McCracken, Daniel, 168
McCullough, Paul, 268
McGonagle, David J., 156
McKinsey, 15
McKinsey & Company, 160
McLuhan, Marshall, 105, 106, 240
McNealy, Scott, 271
meaning, 292
mechanical calculator, 36
mechanical checking, 186
mechanical theorem prover, 64, 70
mechanised proof, 70
mechanised theorem proving, 70
media theory, 18, 107
media,personal dynamic, 19

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

medical systems, 90
medium, 126, 250–254
  for communication, 19
medium for communication, 107
medium for thinking, 18
memex, 88
memory address, 116
memory dump, 144
memory location, 116
memory management, 219, 264
memory region, 230
memory safety, 219
Menkman
  Rosa, 22
Menlo Park, 100, 114
Merry-Shapiro, Diana, 20
message, 250, 261
message sending, 236, 255
messaging, 257
meta-circularity, 87
meta-class, 278
meta-language, 51
meta-medium, 78, 107, 237
meta-object protocol, 279
metalanguage, 214
metaphor, 240
metasyntactic variable, 296
meter, 229
method, 298
methodological incommensurability, 298
methodology, 174
methods
  rigorous, 16
Meyer, Bertrand, 264
Micro-Soft, 115
Microalgol, 51, 71
microcoded CPU, 67
microcomputer, 13, 19, 80, 112–122, 301
microcomputer industry, 119
microcomputer kit, 113
microfilm, 18, 88
microphysics, 272, 291
microprocessor, 67
Microsoft, 228
Microsoft Tay, 28
Microsoft Visual C++ 4.0, 227
Microsoft Word, 119
microworld, 19, 98
Middle Ages, 62
middle school, 107
middleware, 271
MIDI, 124
military, 36, 37, 83, 88, 274
military research, 241
military software, 170
Mills, Harlan, 63, 161, 172
Milner, Robin, 65, 213–217
mind-expanding technology, 297
Mindstorms, 99
minicomputer, 179
Minority Report, 54–56, 205
Minsky, Marvin, 97
missile defence, 166–171
missile trajectories, 169
mission-critical systems, 34
MIT, 3, 11, 14, 77, 81, 83, 88, 168, 180, 209, 248, 263, 278, 290, 301
MIT Artificial Intelligence Lab, 97
MIT Computation Centre, 91
MIT hackers, 24, 126, 144
MIT Lincoln Laboratory, 84, 132
MIT Media Lab, 125
MIT Research Laboratory of Electronics, 77, 84, 144
MIT Tech Square, 12
MITS, 113
mixin, 300
ML (meta-language), 65
ML paradigm, 225, 260
ML type system, 217
ML under UNIX, 216
ML under VMS, 216
mlchanges.doc, 216
mode, 191, 197–199, 205, 212
mode constructor, 205
mode declarer, 205
modelling, 237, 241, 260
modelling language, 260
modernity, 187
Modular Programming Language, 209, 212, 210
modularity, 255
module, 273
module system, 193
modus ponens, 86
molecular biology, 240
Mondrian, Piet, 96
monitoring, 16
Monte Carlo Compiler, 241
Monte Carlo method, 241
moon, 67
Moon, David, 278
Moore School of Electrical Engineering, 39
Moore, J. Strother, 65
Mordor, 24
morph, 262
Morris, James H., 209–212
MOS 6502, 123
Mosses, Peter, 52
mouse, 100
multidisciplinary team, 179
music instrument, 124
music synthesis, 81
music synthesiser, 253
musical actor, 122

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

musical instrument, 20
mutation testing, 62, 159
Myers, Glenford, 157

naive realism, 111
Nake, Frieder, 96
napkin, 276
NASA, 15, 68, 101
Nash, John, 149
National Bureau of Standards, 165
National Institutes of Health, 90
National Security Agency, 157
NATO 1969 Software Engineering conference, 57
NATO Advanced Study Institute, 244
NATO Conference on Software Engineering, 16, 290
NATO Science Committee, 148
NATO Software Engineering conference, 55, 60, 133, 148, 153, 156, 160, 167, 174, 183, 276, 301, 304
NATO Study Group on Computer Science, 148
natural deduction, 195
natural language, 98, 285
Naughton, Patrick, 271
Naur, Peter, 47, 175
NBS, 165
NDC conference, 290
nested boxes, 111, 261
Netscape Communicator 5.0, 305
network, 219
neural network, 4
neurophysiological research, 90
New Flavors, 278
New York Stock Exchange, 5
New York Times, 59
Newcastle University, 155
Newell, Allen, 65, 86
ninth floor, 12
NLS, *see* oN-Line System
No Silver Bullet, 173
Noll, A. Michael, 96
non-computer-based testing, 158
non-hacker, 294
non-linear interaction, 173
Nonaka, Ikujiro, 178
Nordström, Bengt, 222
normal operational range, 170
normal science, 217
Norway, 260
Norwegian Computing Center, 241
notebook, 19
NoteTaker, 108, 254
noun, 238, 274, 275
nuclear launch site, 168
nuclear reactor, 241
nuclear strike, 14, 162
nuclear weapons, 170
numerical analysis, 25
numerical codes, 41
numerical operator, 198
Nygaard, Kristen, 241–246, 260

Oak programming system, 271
Oakland, 122
obfuscation, 295
object, 236, 241, 245, 250, 257, 262, 273, 286, 287, 302
object browser, 108
Object Management Group (OMG), 271, 276
object member, 226
Object Modeling Technique (OMT), 276
object of beauty, 10
Object Oriented Zoned Environment, 253, 256
object-oriented design, 238, 274
object-oriented language, 219
object-oriented programming, 80, 89, 108, 204, 226, 236, 254, 256, 264, 267, 301
Object-Oriented Programming, Systems, Languages and Applications, *see* OOPSLA Conference
Object-Oriented Software Engineering (OOSE), 276
object-oriented system, 182
object-to-think-with, 98
obscurantism, 124
office of the future, 79, 105, 112
office system, 301
ON construct, 152
on-line debugging, 144, 172, 179, 184
oN-Line System, 77, 100, 109
OOPSLA conference, 179, 182, 239–241, 273, 278
OOZE, *see* Object Oriented Zoned Environment
open-source, 186
open-source software, 305
operating system, 116, 174
  OS/360, 152, 171
  UNIX, 3, 12
  verification, 67
operational specification, 162
operations research, 241
operator, 11, 37
OPS-3 programming system, 209
oral tradition, 185
order codes, 140
order of magnitude improvement, 173
organ, 240
orthogonal design, 205
orthogonality, 296
OS/360, 152, 171, 174
oscilloscope, 268
over-engineering, 16, 132, 136, 154
ownership type, 219

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

P-programs, 174
package, 264
pair programming, 17, 124, 181, 182
pair type, 221
Palo Alto Research Centre, *see* Xerox PARC
Panel on Computing in Support of Battle Management, 169
paper record, 191, 299
Papers We Love, 16
Papert, Seymour, 97, 105
paradigm change, 177
paradigm shift, 94, 279
paradigmatic achievement, 1, 291
paradox, 191, 196
parameter testing, 162
PARC, *see* Xerox PARC
PARC Papers for Pendery and Planning Purpose, 112
ParcPlace Systems, 254, 269, 271
Paris, 50
Parnas, David L., 169, 173, 175, 210
participatory design, 134, 163, 177
Passmore, Grant, 6
past, 285
past idea, 304
pattern, 261
pattern of interaction, 287, 300
Paulin-Mohring, Christine, 223
payroll, 200, 202
payroll system, 160, 182
PDP-1, 12, 81–83
Peek and poke statements, 116
peer-review, 286
Pendery, Don, 112
Penrose, Roger, 206
performance, 21, 124, 268
performance assessment, 149
performance overhead, 288
Perlis, Alan, 60, 150, 156
Perlman, Radia, 20
personal computer, 118, 180, 225, 246, 247
personal computer for children of all ages, 78
Personal Dynamic Media, 19, 106–109, 117, 120, 255
personal knowledge, 148
philosophy, 18, 110
philosophy of science, 286, 290
physical law, 168
physical machine, 68
physical system, 241
physical unit, 229
physics, 14
physics textbook, 111
Piaget, Jean, 105
PICTURE clause, 200, 299
PILOT, 92–95
PILOT system, 150
Ping-Pong balls, 36
Pixar Animation Studios, 105
Plan to Throw One Away, 163, 175
planning, 14
pleasantness problem, 22, 175
plotter, 96
pluralism, 23, 28, 73, 193, 194, 291, 298–301, 305
pluralistic discipline, 72
poet, 295
poetic code, 295
pointer, 192, 206, 242
political act, 23
political pamphlet, 62
political struggle, 250
polyhedra, 212
polymorphic function, 218
polymorphic recursion, 220
polymorphism, 215, 229
Polymorphism newsletter, 216
POPL symposium, 62, 153, 255
Popular Electronics, 113
portability, 300
portability problem, 33
Post production systems, 34
post-condition, 48, 57, 265
postmortem dump, 132, 147, 151
postmortem dump routine, 142
Power Peg, 5
power set, 203
power structure, 302
powerful idea, 99
powerful ideas, 26
practical constraint, 212
practical expertise, 286
practical problem, 293
practice, 39, 292
practices
  rigorous, 16
Pragnell, Mervyn, 206, 212, 213
pre-condition, 48, 57, 265
pre-scientific era, 217, 287
predicate, 196
predicate logic, 221
predictability, 177
predictive policing, 24
prefixing, 245, 252, 256
prevention-oriented period, 166
primitive type, 191, 203, 229
Principia Mathematica, 196
Principia Mathematica, 65, 86
principles of software engineering, 166
printf function, 224
privacy freaks, 181
private arcane matter, 184
private communication, 248

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

problem exploration, 181
problem of correctness, 186
problem of pleasantness, 186
procedural attachment, 256
procedure division, 201
process, 243, 294
process model generator, 177
process of refinements, 222
process specification, 286
process-oriented approach, 134, 177
processor, 259
product, 192
product assurance, 163
product development, 179
product-oriented approach, 134, 177
production, 14, 15
production mode, 148
Production of Large Computer Programs, 162
production software, 167
profession, low-status, 40
professional book, 294
professionalisation of programming, 33–36
professionalisation of testing, 183
professionalism, 16, 23
profit leader, 161
program, 67
  correctness, 35
  error-free, 21
  proofs, 60–71
  reasoning, 56–58
  understanding, 23–25
program checkout, 143, 184
program construction, 289
program correctness, 10, 33, 47, 60, 156, 175, 226
program description, 200
program error, 21, 23, 99
program proof, 11, 47
Program Test Methods, 156
program testing, 61
program verification, 61, 63, 167, 290
programmer, 39, 151, 288, 289
  individual, 16
  individual skills, 16
  shortage, 33
programmers, 14
  shortage, 39
programming, 285, 292, 304
  as a craft, 72
  as a mathematical activity, 46
  as a musical instrument, 20
  as a political act, 23
  as translation, 34
  black art, 16
  complexity, 16
  education, 25–26
  engineering discipline, 16
  knowledge, 25
  respectable discipline, 33
  scientific way, 16
  styles, practices and tools, 17
programming by demonstration, 110
programming entity, 299
programming environment, 77, 180, 261, 262
programming error, 227
programming experience, 262
programming gap, 39
programming industry, 156
programming language, 42, 72, 155, 198, 199, 279, 287, 300
  Ada, 153, 184, 237, 263, 274
  Agda, 224
  Algol, 3, 11, 33, 49–54, 71, 109, 202, 226, 242, 248, 304
  Algol 60, 191, 197–199, 290
  Algol 68, 204–205, 212
  Algol W, 191, 212
  Alphard, 256
  as a boundary object, 44
  BASIC, 80, 115–118
  BCPL, 265
  BETA, 260
  C, 3, 12, 295
  C with Classes, 265
  C++, 226, 237, 239, 265–267, 302
  Caml, 224
  Cayenne, 224
  Clu, 153, 193, 211, 256, 263
  COBOL, 44, 191, 200, 212, 304
  COMTRAN, 199–201
  Dependent ML, 224
  Eiffel, 237, 263–265
  Epigram, 224
  Erlang, 185
  EULER, 248
  F#, 65
  FLOW-MATIC, 191, 199–201
  FLPL, 87
  FORTRAN, 16, 115, *see also* QUICKTRAN
  GEDANKEN, 208
  Haskell, 219, 230
  HOPE, 216
  HyperTalk, 121
  Idris, 224
  International Algorithmic Language, 198
  IPL, 86
  ISWIM, 208, 214
  Java, 77, 153, 239, 271–272
  JavaScript, 194, 228, 263
  LISP, 45
  Lisp, 77, 86–87, 92–95, 97, 132, 146–147, 179–181, 184, 278, 290, 300
  LISP 1.5, 249

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

Logo, 19, 26, 95, 97–100, 124, 248, 295
Mesa, 209
ML, 65, 213–217, 304
Objective-C, 269
OCaml, 65
PAL, 208
Pascal, 205, 212
Pharo, 261
PL/I, 152, 184
Rust, 219
Scheme, 95
Self, 261–263
semantics, 49–54
SIMSCRIPT, 242
SIMULA, 203, 226, 241–246, 251, 265
SIMULA 67, 236, 241–246
Smalltalk, 106–109, 182, 236, 239–241, 246–255, 265, 267–271, 286, 300
Smalltalk-74, 254
Smalltalk-76, 255
Squeak, 239, 261
Standard ML, 216
the idea of, 8
TypeScript, 194, 228, 304
programming language concept, 193
programming language semantics, 206–209, 214
programming language theory, 70
programming languages, 125
programming methodology, 54, 136
programming paradigm, 256, 273
programming style, 9, 250, 252
programming system, 94, 110, 179–181, 227, 248, 262, 279, 291
A-0, 34
A-2, 42
Autocode, 34
Boxer, 109–112
FLOW-MATIC, 42
Pygmalion, 109–112
Short Code, 42
programming systems, 125
programming vs. coding, 41, 137
programming, professionalisation of, 33–36
programs
as mathematical entities, 34
progress, 194
Project MAC, 92, 209
project management, 136
project planning, 182
project progress, 165
proof, 49, 64, 66, 193, 212
mathematical, 61
program correctness, 61
reflection, 67
proof assistant, 11, 220–226, 265
proof checker, 214
proof checking, 36, 64
proof of correctness, 67, 156
proofs of programs, 60–71
proper engineering discipline, 161
property inheritance, 256
property list, 249
proposition, 196, 221
Propositions as Types, 195
protection mechanism, 192
prototype, 261
prototyping, 176
psychedelic drugs, 104
psychoacoustics, 88, 101
psychologist, 87
psychology, 291, 293
publication, 300
publication language, 10, 298
pulse-by-pulse, 139
punched card, 11, 41, 81, 214
purpose of testing, 157
Pygmalion, 109–112, 127, 253

Quake III, 295
quality assurance, 132, 178
quality control, 64
quantifier, 221
QUICKTRAN, 145, 191, 197–199

radar, 12, 13
Radiation Laboratory, 12
radio, 140
Radio Shack TRS-80, 115
RAND Corporation, 14
Randell, Brian, 155
range failure, 154
rapid feedback, 182, 238
rate of change, 23
Rational Machines, 274
Rational Software, 274, 276
Rational Unified Process (RUP), 276
Read-Eval-Print Loop, 80, 146, 180, 225
readability, 295
real number, 199
real world, 241, 289, 293
real world effect, 67
real-world software, 63
real-world systems, 167
real-world user, 266
reasoning, 256
reconstructible medium, 111
record, 199–206, 244, 300
record class, 245
record data type, 200
record pointer, 120
record type, 191, 212
recording, 286
recovery block, 155

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

recursion, 209
recursive definition, 203, 206
recursive function, 51
recursive function theory, 45
recursive structure, 192
refactoring, 183
reference, 126
reference counting, 244
reference type, 245
reflection, 227
reflective capabilities, 227
refrigerator, 67
region-based memory management, 219
reinforcement learning, 97
reliability, 132, 263
reliable hardware, 170
Remington Rand, 199
remote accessing, 243
repetitive structure, 171
REPL, *see* Read-Eval-Print Loop
representation, 210
representation theorem, 211
reproducibility, 177
reproduction study, 49
require clause, 265
requirements, 22, 133, 156
requirements definition, 178
requirements gathering, 175, 238
research paradigm, 217
research retreat, 108
Responsibility-Driven Design, 269, 273, 297
reusability, 264
reuse, 263
Revised report on the algorithmic language Algol 60, 50
revolutionary change, 99
rewrite rule, 225
Reynolds, John C., 193, 211
right to explanation, 4
rigorous argument, 134
robot, 42
  turtle, 19
Rolling Stone, 104, 251
Rome, Italy, 56
Ronald Reagan, 169
Royal Signals and Radar Establishment, 69
Royce, Winston, 134, 163, 175
rugby, 178
Rule 15c3-5, 7
Rule of Three, 288
Rumbaugh, James, 276
runtime check, 242, 299
Russell's paradox, 196
Russell, Bertrand, 196, 206
Russell, Steve, 81, 87, 91
Russia, 23
S-expression, 203
S-programs, 174
Safeguard system, 167
safety, 194, 228, 256
SAGE, 13, 41, 77, 84, 88, 167
SAIL, *see* Stanford AI Laboratory
Sammet, Jean, 245
Samson, Peter, 83
San Francisco, 100
scale, 240
scale of software, 238
scaling-up, 173
Scandinavia, 177, 260
Scandinavian approach, 177, 181
Scandinavian school, 260
Scheme, 95
schoolchildren, 124
Schopenhauer, Arthur, 240
Schwaber, Ken, 179
science, 289, 291
Science and Engineering Research Council, 216
scientific code, 295
scientific discipline, 286
scientific enterprise, 183
scientific method, 16, 17, 133, 147, 149
scientific paradigm, 9, 217, 285
scientific pluralism, 28
scientific practice, 49
scientific programming, 152, 200, 202
scientific research paradigm, 46, 290
scientific way, 16
scientist, 295
Scott, Dana, 212, 214
Scratch, 112, 125
screen turtle, 98
SCRUM, 134
SCRUM methodology, 179, 181
SDC, 14
SDI, 169
seal and unseal operations, 211
SECD machine, 208
secrecy problem, 210
Securities and Exchange Commission, 5, 7
sel4 microkernel, 3, 11
self, 261
self-checking, 155
self-consciousness, 96
self-modifiability, 237
self-referential proposition, 198
semantic incommensurability, 298
semantics, 49–54, 242
semantics of programming languages, 213
Semi-Automatic Ground Environment, 13, 84, 161
separation logic, 65
SERC, 216
serendipity, 99, 125
set, 191, 192

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

set theoretical model, 206
set-theoretic model, 210
SHARE, 43
SHARE user group, 201
shared entity, 300
shared memory, 219
Shaw, Clifford, 86
Short Code, 41
Short-Range Committee, 200
Shoup, Richard, 105
Show us your screens, 124
Simon, Herbert, 65, 86
Simondon, Gilbert, 24
Simonyi, Charles, 119
simplicity, 295
simplification algorithm, 214
simply typed lambda, 209
simpterm function, 214
simulation, 258, 277
simulation laboratory, 262
Simulation Language, 241
Single E.P, 141
Sketchpad, 19, 89, 101, 105, 247, 297
Slug, *see* Russell, Steve
Smalltalk, 19
Smalltalk-72, 108, 110, 117
Smalltalk-72 handbook, 251
Smalltalk-76, 108
SMARS, 5
Smith, Alvy Ray, 105
Smith, Brian Cantwell, 67
Smith, David Canfield, 110, 119
Smith, E. E., 81
Smith, Randall, 261
snapshots, 47
Snyder, Betty, 36
social activity, 293
social control, 193
social history, 303
social process, 35, 60–63, 66, 70, 72, 133, 186, 194, 218
Social Processes and Proofs of Theorems and Programs, 61
social structure, 23
social transmission, 292
societal impact, 305
societies of high modernity, 186
society, 18, 96
socio-economic factor, 302
socio-political context, 99, 287
socio-technical issues, 24
SofTech, Inc., 153
software
  complexity, 23
  error-free, 21–23
  financial, 6
  opacity, 23
software engineering, 55
Software aspects of strategic defence systems, 169
software component, 64
Software Configuration Control Board, 15
software correctness, 61
software crisis, 293, 303
software developer, 295
software development, 288
software development lifecycle, 238
software development methodology, 167, 186
software development process, 72
software engineer, 182, 294
software engineering, 28, 64, 80, 149, 151, 156, 167, 178, 237, 261, 263, 289, 301, 303
Software Implemented Fault Tolerance, 67
Software Initiative, 153
software lifecycle, 301
software manufacturing, 147
software production, 55
software projects, 59
software quality, 64
Software Quality Engineering, 183
software reliability, 155
software structure, 293
software systems, 166, 175
Software Technology for Evolutionary Participative System Development, 177
software testing, 156–159, 183
software verification, 49, 70
Solomon, Cynthia, 20, 97, 98
Sonic Pi, 3, 20, 124
soundness, 228
soundness property, 211
source code, 78, 180, 294
Soviet Union, 14
Spacewar!, 3, 81–83
Spacewar article, 251
spatial metaphor, 111
special form, 258
specification, 14, 22, 64, 67, 133, 149, 174, 175, 194, 223, 285
Sperry Rand corporation, 208
spiral development model, 176, 275
spiral model, 179
spiral software lifecycle, 274
spreadsheet, 80, 119
sprint, 179
standard, 294
Standard ML, 216
standardisation, 156
Stanford, 180, 213
Stanford AI Laboratory, 180
Stanford LCF, 214
Stanford Research Institute, 100
Star Wars program, 169
state transition, 275

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

static type system, 215
station, 243
Steele, Guy, 95
STEPS, 177
Stockham, Thomas, 144
storage location, 141
stored program, 141
Strachey, Christopher, 52, 206, 212
Strategic Defense Initiative, 169
Stroustrup, Bjarne, 226, 265–267, 275, 301
structure editor, 93, 109, 117, 146
structured data, 200
structured programming, 17, 35, 57, 58, 72, 136, 149, 161, 193, 203, 242, 243, 259, 275, 290, 300
Structured Programming Theorem, 58
struggle for control, 301
subclass, 245, 255, 256
subculture, 289, 292, 295
subroutine, 116, 141
subroutine library, 160
subscripted variable, 199
subtyping, 236
sufficiency, 209
Sun Microsystems, 271
Sun workstation, 271
supervisor process, 185
supporting programmer, 59
surgical team, 64, 172
Sussman, Gerald, 95
Sutherland, Bert, 89, 267
Sutherland, Ivan, 19, 89, 246
Sutherland, Jeff, 179
Sven, Doug, 6
Swansea, 213
switching theory, 25
symbiotic system, 94
symbol manipulation, 86
symbol-processing, 301
syntactic structures, 68
syntax, 50
syntax directed interpreter, 258
Sys statement, 116
SYSDEFS file, 252
system availability, 149
system decomposition, 210
System Development Corporation, 103
system of expertise, 24
system of practice, 285
system test, 149
system testing, 165
System/360, 171
systems
  mission critical, 34
  technically challenging, 16
Systems Development Department, 112
systems of practice, 298

tablet, 106
tacit knowledge, 26, 39, 286, 294, 299
tactic, 215, 225
Takeuchi, Hirotaka, 178
tape, 201
tape reader, 113
tar pit, 159
tax return, 173
Taylor, Bob, 101
TDD, *see* Test-Driven Development
team, 293
team role, 182
Team Topologies, 59
Tech Model Railroad Club, 12, 77, 81, 85
technical artifact, 23, 265
technical concept, 8, 185
Technical University of Berlin, 177
Technical University of Munich, 148
Teitelman, Warren, 94, 150
Tek 4404, 269
Tek Labs, 268, 273
Tektronix, 238, 268, 272, 278
tele-conferencing, 100
telecommunications, 185
telephone exchange, 185
teletype, 19, 113
teletype terminal, 117
temperature, 67, 289
template meta-programming, 220
TEN-Mode, 92
tentative context, 268
terminal, 19, 82, 91, 180
terminology, 256
test, 133, 184, 191, 286
test automation, 150
test documentation, 165
test manager, 156
test selection, 157
test technician, 156, 183
test trajectory, 139
Test, then code, 166, 183
Test-Driven Development, 17, 80, 133, 139, 183, 184, 186, 269, 272, 297, 301
testing, 16, 17, 61, 62, 132, 133, 136, 149, 156–159, 175, 182, 238, 300
testing practices, 156
testing tools, 156
testing,, 23
tests
  as a specification, 183
text editing, 100
text editor, 248, 253
text processing, 119
Thacker, Chuck, 105
The Art of Computer Programming, 40

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

The Art of Software Testing, 157
The Atlantic, 18, 294
The Debugging Paradox, 184
The Duke of Marlborough, 207
The Growth of Software Testing, 156
The Hub, 123
The League of Automatic Music Composers, 123
The Mother of All Demos, 77, 100, 286, 294
The Mythical Man-Month, 169, 172
The New New Product Development Game, 178
The Principles of Mathematics, 196
The Telekommunist Manifesto, 23
theorem, 62, 65
theorem prover, 63–66, 213, 220–226
  ACL2, 6
  ALF, 222
  Automath, 220, 222
  Boyer-Moore, 65, 66, 213
  Coq, 223
  LCF, 193, 220
theorem proving, 213, 300
theoretical computer science, 191
theoretical curiosity, 166
theory, 260
theory of computation, 202
theory of data types, 203
Therac-25, 1
ThingLab, 262
thinking
  in children, 97
  in machines, 97
thm type, 215
Three Amigos, 276
throw function, 154
time travel debugging, 133
time-sharing, 56, 77, 91–92, 97, 103, 115, 132, 145, 172, 179–181, 214, 225, 247
timing, 67
Titanic effect, 36
TMRC, *see* Tech Model Railroad Club
Toggle Switch Storage, 84
tools, 16, 226
tools for thought, 297
top management, 161
top-down development methodologies, 161, 175
top-down software development, 59
TOPLAP, 20, 124
trade union, 260
trade-off, 228, 295
trading system, 5
trading zones, 60, 230, 292
training costs, 263
transfer instruction, 141
Tree-Meta, 102
tricks, 141
tripartite nature, 289
trust, 185–187
trustworthiness, 24
trustworthy software, 304
Turing Award, 60
Turing machines, 34
Turing, Alan, 141, 143
Turtle graphics, 19, 98, 109, 111
TX-0, 12, 77, 81–86
TX-0 tools, 85
TX-2, 19
type, 212, 286, 287, 298, 300
  dependent, 193
  fixed point, 197
  floating point, 197
  integer, 197
  nature, 197
  primitive, 191
  unsound, 194
type annotation, 210
type assignment, 210
type checker, 218, 219
type checking, 194, 210, 222
  compile-time, 193
  run-time, 193
type classes, 220
type error, 227
type inference, 209, 210, 219
type inhabitation, 221
type system, 218, 226, 264, 299, 304
type theory, 222
type-systems laboratory, 220
types
  as sets, 192
  formal origins, 196–197
Types and Programming Languages, 218
types as sets, 229
Types course, 226

UC Santa Barbara, 264
UK Ministry of Defence, 69
UK Ministry of Technology, 183
UML, *see* Unified Modelling Language
unchecked type conversion, 266
Uncle John, 290
undecidability, 166
Underestimates and Overexpectations, 168
underground logic reading group, 206, 216
underground reading group, 213
understanding, 295
unexpected situation, 151
Ungar, David, 261
unification, 252
Unified Modelling Language (UML), 239, 276
unified paradigm, 298
union of non-intersecting sets, 203
union type, 221
unit test, 149
unit testing, 15, 165, 183

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010

USA, 263
United States Army, 37
United States Military, 83
units of measure, 229
Univac, 42, 242
UNIVAC I, 41, 199
universal programming language, 43, 202
universal quantification, 221
universe of values, 210
university, 10, 13
university computing centre, 25
university department, 292
University of Cambridge, 38, 69, 226
University of North Carolina, 156, 173
University of Oxford, 52
University of Utah, 247
University of Vienna, 177
UNIX, 3, 14
Unlocking the Computer's Profit Potential, 106, 160
unsound type system, 228
unwritten knowledge, 220
US Air Force, 247
US Air Force Academy, 274
US Census Bureau, 39
US Department of Defense, 153, 165, 237, 263
US military, 171
US National Academy of Engineering, 266
US Navy, 100
USA, 263
Use Case diagram, 239
user, 78
user-defined types, 205, 267
user interface toolkit, 272
UT-3, 84, 144
Utility Decks, 92
Utility Tape 3, 84, 144

vacuum tubes, 38, 84, 139
validation, 165
variable, 261
variant record, 205, 213
VAX machine code, 216
VDL, *see* Vienna Definition Language
vector, 224
verb, 238, 274, 275
verification, 149, 165
verification debate, 11
Vernacular of File Formats, 22
very-high-level programming language, 258
vicious-circle principle, 196
Victor, Bret, 297
video display, 180
video switch, 180
video terminal, 113
Vienna Definition Language, 53
Vienna Development Method, 135
Vienna IBM laboratory, 53
vim editor, 226
VIPER microprocessor, 69
virtual machine, 268, 271
virtual memory, 253
virtual method, 245
VisiCalc, 80, 119
visual effects, 81
visual element, 262
Visual Smalltalk Enterprise, 239, 270
VisualWorks, 270

Wadler, Philip, 195
Wadsworth, Christopher P., 52
warfare, 67
water flow, 137
Waterfall, 134, 149, 161–166, 179, 274
web browser, 271
Wegner, Peter, 259
Weizenbaum, Joseph, 28, 135, 168
well-typed program, 193, 209, 211, 216, 217
West coast, 13, 78, 100
West Coast Computer Faire, 267
Western annual Joint Computer Conferences, 201
Wheeler, David, 141
while statement, 57
Whirlwind, 12, 77, 83–86
Whitehead and Russell, 86
Wiitanen, Wayne, 81
Wilkes, Maurice, 38, 140–141
Winograd, Terry, 256
Wirfs-Brock, Allen, 268, 273
Wirfs-Brock, Rebecca, 273
Wirth, Niklaus, 151, 205, 248
WJCC, 201
women
  exclusion from the workforce, 33
  in the hacker culture, 12
  in the humanistic culture, 20
word processing, 112, 172
World War II, 83, 88
World Wide Web, 88, 272
Wozniak, Steve, 115
WYSIWYG, 119

Xerox Alto, 106, 112, 267
Xerox PARC, 67, 79, 104, 106, 126, 180, 209, 210, 212, 247, 250–254, 267, 277, 301
Xerox Star, 79, 112, 119, 127
XINTF function, 198
XP, *see* Extreme Programming
XP customer, 182

Y2K bug, 143
Yale University, 61

Zen and the Art of Motorcycle Maintenance, 140
Zilles, Stephen, 211, 256
Zurich, 50

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 16 Nov 2026 at 23:44:46, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/9781009492379.010