

Index

A

- `add()` method, JavaScript
 - DOMTokenList, 265
- `addEventListener()` method, JavaScript, 268–269
- `after()` method, JavaScript, 261
- AI *see* artificial intelligence
- American Standard Code for Information Interchange (ASCII), 12
- anagram guessing game (Python project), 180
 - completing a round, 189
 - displaying welcome message, 184
 - finding anagrams, 183–184
 - full code of, 189–193
 - getting a random word and its anagrams, 186
 - handling user input, 187–188
 - loading a list of words from a CSV file, 180–183
 - looping a single round, 187
 - setting up game loop, 184–185
 - tracking user data, 186
- Andreesen, Marc, 19
- Android app development, 63
- anonymous functions, JavaScript, 246
- API List, 170
- APIs *see* application programming interfaces
- `append()` method
 - JavaScript, 261
 - Python, 108–109
- application development, 54
 - Android, 63
 - web, 63
 - Windows, 60
- application programming interfaces (APIs), 61, 65, 142
 - errors, 173–174
 - getting random quotation from (JavaScript project), 286–292
 - Python, 165–170
- arithmetic assignment operators, 31, 94, 95
- arithmetic operators, 30–31
- array literals, 244
- arrays, 26
- arrays, JavaScript
 - declaring, 244–245
 - for `. . . of` loop, 247
 - `forEach()` method, 245–247
 - length property, 248, 256
 - methods, 248, 249
- artificial intelligence (AI), 1, 20, 54, 295
 - avoiding common mistakes when using, 301
 - best practices for integrating, 300–301
 - and coding, 296, 298

- artificial intelligence (AI) (*continued*)
 - coding tools, 299–300
 - dependency, avoiding, 298–299
 - prompting, 297–298
 - strengths and limitations of coding assistants, 296–297
 - understanding code generated by, 301–302
 - vibe coding, 300
- ASCII *see* American Standard Code for Information Interchange
- assembler, 52
- assembly language, 50–53
- assignment operators, 29, 90
- async function, JavaScript, 288
- asynchronous operations, 288–289
- attribute selector, 310
- automation, 20, 54, 65, 296
- await operator, JavaScript, 289, 290

B

- back end, 21, 55, 56, 61, 65, 288
- backtick, 233–234
- before() method, JavaScript, 262
- big data, 3
- bit, 12
- block scope, JavaScript, 228
- block syntax, 34, 35
- blockchain, 66
- bool() function, Python, 93
- Boolean
 - literals, 29
 - Python, 90, 91, 92, 115, 121
- Booth, Kathleen, 51
- bootloader, 8

- break mode, 45, 46, 275, 276
 - breakpoint, 276–277
 - exiting, 278
 - using debugger statement for entering, 277
 - viewing a variable value in, 278
 - see also* stepping (debugging)
- break statement
 - JavaScript, 223–224
 - Python, 122, 151, 185
- breakpoint, 45
 - removing, 277
 - setting, 276, 277, 280
- buffer overflow, 58
- byte, 12

C

- C, 11, 12, 58–59
- C++, 59–60
- C#, 60–61
- callback function, 245, 246, 268, 269, 270, 285
- camelCase, 25
- Cascading Style Sheets (CSS), 55, 209, 211, 303
 - modification of, 264–267
 - properties, 306–308
 - selectors, 255, 308–312
 - see also* Hypertext Markup Language (HTML)
- central processing unit (CPU), 51, 52
- charAt() method, JavaScript, 236
- charCodeAt() method, JavaScript, 236
- ChatGPT (OpenAI), 299
- child combinator (>), 310

- child element
 - getting parent of, 260
 - new, adding, 260–263
- children property, JavaScript, 258–259
- choice() function, Python, 151, 186
- class, CSS
 - adding to an element, 265–266
 - removing, 266
 - toggling, 267, 285
- class, Python, 159–161
 - attributes, 162, 164
 - building, 161–162
 - class initializer, 161–162
 - methods, 159
 - properties, 159
- class name, specifying DOM elements by, 254–255
- class selector, 309
- class statement, Python, 160, 161
- classList property, JavaScript, 265–266
- Claude (Anthropic), 299–300
- cloud computing, 61, 63, 66
- CMS *see* content management systems
- code blocking, and synchronous operations, 288
- coding, 1, 7, 8–9
 - as a bonus skill, 18
 - and creativity/fun, 16–17
 - and jobs, 18
 - and nerd stereotype, 15–16
 - real-world uses of, 19–21
 - reasons for learning, 15–19
 - and thinking skills, 16
 - three-and-a-half Rs of, 13
 - as a universal language, 17
 - writing and execution of code, 12–13
 - see also* artificial intelligence (AI); JavaScript; Python
- coding concepts
 - comments, 43–44
 - conditionals, 33–36
 - data types, 27–29
 - debugging and handling errors, 44–47
 - expressions, 29–33
 - functions, 38–40
 - loops, 36–38
 - objects, 41–43
 - variables, 24–26
- command line
 - definition of, 72
 - launching Terminal on macOS, 74
 - launching Terminal on Windows, 73
 - and Python, 72–73
 - running Python on Windows, 82, 83
 - tools, 61
- Command Prompt, 73
- comments, 43–44
 - and debugging, 47
 - JavaScript, 212–213
 - Python, 123
 - turning problem statements into, 47
- comparison expressions, 32–33
- comparison operators, 32, 94–95
- compiler, 12, 13, 58, 68
- computers, 8–9, 10, 11–12
- concat() method, JavaScript, 249
- concatenation operator, 32, 93, 99
- conditional execution, 34

- conditionals, 33–34
 - `if. . . else` statements, 35–36
 - Python, 116–119
 - true/false decisions, 34–35
- Console window in web browsers, 215, 216–217, 273
- debugging with, 273–275
- displaying, 274
- executing code in, 274–275
- logging data into, 217, 218, 274
- `console.log()` method, JavaScript, 217, 227, 228, 229, 230, 246, 247, 274
- `const` keyword, JavaScript, 220, 228
- content management systems (CMS), 65
- `continue` statement
 - JavaScript, 224
 - Python, 122–123, 197
- `count()` method, Python, 100
- CPU *see* central processing unit
- `createElement()` method, JavaScript, 261
- creativity, and coding, 16–17
- cross-platform apps, 63
- CSS *see* Cascading Style Sheets
- `csv.DictReader()` method, 181–182
- cybersecurity, 66

D

- data science, 20, 54, 64
- data types, 27
 - Boolean literals, 29
 - conversion, in Python, 92–93
 - mixing, in Python, 91–92
 - numerical literals, 27–28, 90
 - Python, 91–93, 176
 - string literals, 28, 90, 91, 232–234

- date and time, JavaScript, 236–237
 - arguments, 237
 - current, 238
 - extracting information about, 238, 239
 - setting, 239–240
 - specific, 238
- date class, Python, 139–140
- `Date()` function, JavaScript, 238
- `datetime` module, Python, 135, 139–140
- debugger statement, JavaScript, 277
- debugging
 - Python code, 174–176
 - strategies, 46–47
 - techniques, 45–46
 - tools, in web browsers, 214, 215, 272, 273
 - use of AI for, 296
 - and variable scope, 132
- debugging, JavaScript, 271
 - with Console window, 273–275
 - pausing the code, 275–279
 - stepping, 279–281
 - tools, 272–273
- decrement operator, 31
- descendant combinator (space), 310
- DevOps, 61
- dictionary, Python, 113–114, 168, 181–182
- distributed systems, 61
- Django, 20, 55
- `do. . . while` loop, JavaScript, 222–223
- Document Object Model (DOM), 252–253
 - getting data about event, 269–270
 - listening for event, 269, 270
 - manipulation of elements, 260–263
 - modification of CSS, 264–267

- specifying elements in, 253–257
- traversing, 257–260

DOM *see* Document Object Model

dot notation, 42, 136, 143, 164

.NET framework., 60

E

ecommerce websites, 65

elements, Document Object Model, 252

- adding class to, 265–267
- adding new element as a child, 261–262
- adding text and tags to, 262–263
- creating new element, 260–261
- getting the children of a parent element, 257–260
- getting the parent of a child element, 260
- modification of CSS, 264–267
- removing, 263
- removing class from, 266
- specifying by class name, 254–255
- specifying by ID, 254
- specifying by selector, 255
- specifying by tag name, 254–255
- styles, changing, 264
- toggling class, 267, 285
- working with collection of, 255–257

`elif` statement, Python, 117–119, 188

embedded systems, 58, 59, 66

empty string *see* null string

`endsWith()` method, JavaScript, 235

`endswith()` method, Python, 100

equal to operator, 29

errors, code

- JavaScript, 291–292
- types of, 45–46
- see also* debugging

errors, Python

- API errors, 173–174
- common types of, 172
- debugging, 174–176
- decoding traceback messages, 170–171
- try and except blocks, 171–174

events, 42, 224, 267–268

- event handlers, 267–268
- event listener, 268, 287
- getting data about, 269–270
- listening for, 268–269, 270

except block, Python, 171–174, 181, 195

`exists()` function, Python, 151

exponential notation, 27–28

expressions

- comparison, 32–33
- debugging, 47
- definition of, 29
- logical, 33
- numeric, 30–31
- Python, 93–97
- string, 32

F

`fetch()` method, JavaScript, 289–290

files, Python, 144

- opening, 145
- reading data from, 146–147
- writing data to, 145–146

- filter, in Python list
 - comprehensions, 156–157
- find() method, Python, 100
- Firefox, 215, 272, 274
- firmware, 8
- :first-child pseudo-class, 311
- Flask, 20, 55
- float() function, Python, 93
- float keyword, JavaScript, 25
- floating-point numbers, 25, 27–28, 240, 241–242
- for . . . of loop, JavaScript, 247, 256
- for loop
 - JavaScript, 221–222
 - Python, 112, 119–121, 128, 147, 198
- forEach() method, JavaScript, 245–247
- f-string (formatted string), 99, 128
- full-stack developers, 64
- function scope, JavaScript, 229
- functions, 38–40
 - Python, 128–134
 - and stepping, 45–46, 280–281
- functions, JavaScript, 224–225
 - calling, after page is loaded, 226–227
 - calling, when <script> tag is parsed, 225–226

G

- game development, 21, 54, 57, 58, 59, 63, 66
 - see also* anagram guessing game (Python project)
- Gemini (Google), 300
- get() method, Python, 167

- getDate() method, JavaScript, 239
- getDay() method, JavaScript, 239
- getElementById() method, JavaScript, 254
- getElementsByClassName() method, JavaScript, 254, 255
- getElementsByTagName() method, JavaScript, 254, 255–257
- getFullYear() method, JavaScript, 239
- getHours() method, JavaScript, 239
- getMilliseconds() method, JavaScript, 239
- getMinutes() method, JavaScript, 239
- getMonth() method, JavaScript, 239
- getSeconds() method, JavaScript, 239
- getTime() method, JavaScript, 239
- GitHub Copilot, 299, 300
- global scope
 - JavaScript, 230
 - Python, 132
- Go, 25, 61–62
- Golang *see* Go
- Google Chrome, 214, 272, 274
 - break mode, 276
 - Console window in, 217, 218
 - HTML viewer in, 216, 273
- graphics, 59

H

- hardware control, 55
- high-level languages, 11, 51, 60
- high-performance applications, 58, 59, 61
- histograms, 202
- :hover pseudo-class, 311

HTML *see* Hypertext Markup Language
HTMLCollection object, JavaScript, 258
HTTP status codes, 167
Hypertext Markup Language (HTML), 55,
209, 210, 211–212, 303
 basic template, 304–305
 file, embedding JavaScript code
 inside, 14–15
 HTML viewer, 215, 216, 272, 273
 tags, 305–306
 see also Cascading Style Sheets (CSS);
 Document Object Model (DOM);
 JavaScript

I

id attribute, 254, 309
ID selector (#), 309
IEEE Spectrum, 50
if . . . else statement, 35–36,
116–117, 188
if statement, 34–35
 JavaScript, 224
 Python, 115–116, 157, 185, 188
import statement, Python, 136, 137,
138–139, 143
in keyword, Python, 110
includes() method, JavaScript, 235
increment operator, 31
indentation of code, 35, 39, 46,
116, 120, 129
index number, 26
indexes, Python
 list, 107–108
 string, 97–98
 tuples, 112

indexOf() method, JavaScript, 235
__init__() function, Python, 160, 162
innerHTML property, JavaScript, 262, 263
input() function
 JavaScript, 224
 Python, 102–104, 116, 185, 187
insert() method, Python, 108–109
instance attributes, 162
int() function, Python, 93, 104
integers, 27, 240–241
interactive mode, Python, 81, 85
Internet of Things (IoT), 21, 55
interpreter, 12, 13, 24, 25, 58
 definition of, 68
 Python, 80–83, 85, 86, 170
IoT *see* Internet of Things
isNaN() function, JavaScript, 224

J

Java, 62–63
Java Virtual Machine (JVM), 62
JavaScript, 9, 41, 55–57, 64, 65, 209–210
 adding comments to code, 212–213
 arrays, 244–248, 249
 basic script construction, 211–212
 coding process, 14–15
 comparison operators, 33
 date and time, 236–240
 debugging, 271–281
 external files, creating, 213–214
 functions, 224–227
 getting a random quotation from an API
 project, 286–292

- JavaScript (*continued*)
 - loops, 221–224
 - math object, 240–244
 - nesting quotation marks in, 28
 - objects, 42, 231–249
 - photo gallery project, 284–285
 - real-world uses of, 20–21
 - strings, 231–235
 - TypeScript, 67–68
 - variable scope, 228–230
 - variables in, 25, 220–221
 - see also* Document Object Model (DOM)
- JavaScript Object Notation (JSON), 168, 290–291
- join() method
 - JavaScript, 249
 - Python, 102, 183
- JSON *see* JavaScript Object Notation
- json() method
 - JavaScript, 290
 - Python, 168
- JVM *see* Java Virtual Machine

K

- key (Python dictionary), 113–114
- key-value pairs (Python dictionary), 113–114
- keywords, definition of, 10
- Kotlin, 63–64

L

- lambda keyword, Python, 198
- lastElementChild property, JavaScript, 259
- lastIndexOf() method, JavaScript, 235

- len() function, Python, 98, 106, 112, 156
- length
 - of arrays, JavaScript, 248, 256
 - sentence, analyzing, 200–202
 - of strings, JavaScript, 234
- let keyword, JavaScript, 220, 228
- lexicon-based sentiment analysis, 203
- libraries, Python, 54, 141–142, 193–194
 - importing and using, 143–144
 - installation of, 142–143
 - for interacting with APIs, 166–167
- list comprehensions, Python, 183–184, 186, 200
 - adding filter, 156–157
 - setting up, 155–156
- lists, 26
- lists, Python, 105–106
 - adding items, 108–109
 - changing an item in, 108
 - constructing, 106
 - getting an item from, 107–108
 - looping through, 120
 - from range of numbers, 106–107
 - removing items, 109–110
 - searching in, 110
- literals
 - array, 244
 - Boolean, 29
 - definition of, 27
 - numerical, 27–28, 90
 - string, 28, 90, 91, 232–234
- local scope
 - JavaScript, 229
 - Python, 133–134

- logic errors, 45, 46
- logical expressions, 33, 95–96
- logical operators, 33, 96
- loop counter, 221–222
- loops, 36–38
- loops, JavaScript, 221
 - do . . . while loop, 222–223
 - execution, controlling, 223–224
 - for loop, 221–222
- loops, Python, 119, 151, 184–185, 196, 197
 - execution, interrupting, 122–123
 - through collection of things, 119–121
 - when a condition is true, 121–122
- lower() method, Python, 100, 185, 195

M

- machine language, 10–11, 50–51
- machine learning, 20, 54, 64
- macOS
 - checking Python installation status in, 76
 - Console window in browsers of, 274
 - installation of Python library on, 142–143
 - installation of Python on, 79–80
 - launching Terminal on, 74, 74
 - running on Windows, 82, 83
 - running Python on, 82, 83
 - running Python script file in, 86
 - Terminal shortcuts, 74–75
 - web development tools in browsers of, 214–217, 272
- main() function, Python, 151
- math module, Python, 135, 138, 139
- math object, JavaScript, 240–244

- Math object methods, JavaScript, 242, 244
 - Math.abs(), 243
 - Math.cbrt(), 243
 - Math.ceil(), 243
 - Math.cos(), 243
 - Math.exp(), 243
 - Math.floor(), 243
 - Math.log(), 243
 - Math.max(), 243
 - Math.min(), 243
 - Math.pow(), 243
 - Math.random(), 243
 - Math.round(), 243
 - Math.sin(), 243
 - Math.sqrt(), 243
 - Math.tan(), 243
 - Math.trunc(), 243
- Math object properties, JavaScript, 242, 244
 - Math.E, 243
 - Math.LN2, 243
 - Math.LN10, 243
 - Math.LOG2E, 243
 - Math.LOG10E, 243
 - Math.PI, 243
 - Math.SQRT1_2, 243
 - Math.SQRT2, 243
- math operators, Python, 94–95
- Matplotlib library, 194, 201
- mean() function, Python, 137–138
- memory leaks, 58, 59, 65
- method chaining, 185
- microservices, 61
- Microsoft Edge, 215, 272, 274

- mnemonics, 51
- mobile apps, 21
- modules, Python, 134–135
 - creating, 140–141
 - element, importing, 138–140
 - importing, 136, 137
 - using, 136–138
- multiline strings, 233, 234

N

- Natural Language Toolkit (NLTK), 193–194, 200, 203
- networking, 61, 66
- newline character, Python, 146
- NLTK *see* Natural Language Toolkit
- Node.js, 21, 65
- :nth-child(*n*) pseudo-class, 311–312
- null string, 28
- numbers
 - floating-point, 25, 27–28, 240, 241–242
 - integers, 27, 240–241
 - making Python list from, 106–107
 - numeric expressions, 30–31
 - numerical literals, 27–28, 90
 - Python, 91
 - and strings, conversion
 - between, 240–242

O

- object-oriented programming (OOP), 59, 60
 - class, 159–161
 - Python, 158–165

- objects, 41–43
 - JavaScript, 231–249
 - Python, 163–165
- online data sources, 54, 57
- OOP *see* object-oriented programming
- open() function, Python, 145, 146, 181, 195
- operands, 30
- operating systems, 58, 59
 - see also* macOS; Windows
- operators, 30
 - arithmetic, 30–31
 - arithmetic assignment, 31, 94, 95
 - assignment, 29, 90
 - comparison, 32, 94–95
 - logical, 33, 96
 - math, 94–95
- order of operations, Python, 96–97

P

- page-level scope *see* global scope
- parent element, 264
 - getting all child nodes, 258–259
 - getting first child node, 259
 - getting last child node, 259
- parentNode property, JavaScript, 260
- parseFloat() function, JavaScript, 241–242
- parseInt() function, JavaScript, 240–241
- PEMDAS, 96–97
- Phaser.js, 21
- photo gallery (JavaScript project), 284–285
- PHP, 64–65

- pip, 142
- + operator, JavaScript, 242
- pop() method
 - JavaScript, 249
 - Python, 109–110
- prepend() method, JavaScript, 261
- print_info() function, Python, 160, 163
- program errors *see* errors, code
- programming *see* coding
- programming languages, 9–10
 - assembly language, 50–53
 - C, 11, 12, 58–59
 - C#, 60–61
 - C++, 59–60
 - compiled, 68
 - Go, 25, 61–62
 - interpreted, 68
 - Java, 62–63
 - Kotlin, 63–64
 - PHP, 64–65
 - ranking of, 50
 - role of, 10–12
 - Rust, 65–66
 - Swift, 66–67
 - Typescript, 67–68
 - see also* JavaScript; Python
- prompt() function, JavaScript, 222, 240
- prompting, AI, 296, 297–298
- PSF *see* Python Software Foundation
- public APIs, 169
- push() method, JavaScript, 249
- PyPI *see* Python Package Index
- pyplot() function, Python, 200–201
- Python, 9, 53–55, 71–72, 179
 - anagram guessing game project, 180–193
 - cat fact cards (example), 176–178
 - coding process, 13–14
 - and command line, 72–73
 - comments, 123
 - conditionals, 116–119
 - data types, 91–93, 176
 - debugging, 174–176
 - declaring arrays/lists in, 26
 - dictionary, 113–114, 168, 181–182
 - expressions, 93–97
 - external libraries, 141–144
 - files, 144–147
 - functions, 128–131
 - handling program errors, 170–174
 - interactive mode, 81, 85
 - list comprehensions, 155–157, 183–184, 186, 200
 - lists, 105–110
 - loops, 119–123, 184–185, 196, 197
 - modules, 134–141
 - order of operations, 96–97
 - program, running, 84–87
 - Pythonic code, 154–155
 - quotations archive (example), 147–151
 - real-world uses of, 20
 - running on Windows, 82, 83
 - script mode, 81–82, 85–87
 - sets, 114–115
 - standard library, 134–135

Python (*continued*)

- strings, 97–104
- survey bot (example), 123–215
- text analyzer project, 193–206
- tuples, 111–112
- variable scope, 131–134
- variables in, 24, 90, 99
- version number of, 75, 82
- see also* libraries, Python
- Python, APIs in, 165–166
 - connection, 166–167
 - handling errors, 173–174
 - repositories, 169–170
 - working with API data, 167–169
- Python, installation of, 75
 - checking installation status, 76–77
 - on macOS, 79–80
 - on Windows, 77–79
- Python, OOP in, 158–159
 - building class, 161–162
 - class, 159–161
 - creating objects, 163–164
 - using objects in code, 164–165
- Python launcher (Windows), 82
- Python Package Index (PyPI), 142
- Python Software Foundation (PSF), 135

Q

- `querySelector()` method, JavaScript, 255
- `querySelectorAll()` method, JavaScript, 255
- quotation marks, in string literals, 28, 90, 232, 233

R

- `raise_for_status()` method, Python, 174
- random module, Python, 135, 151, 186
- random quotation from API (JavaScript project), 286–287
 - asynchronous operations, 288–289
 - await operator, 289
 - `fetch()` method, 289–290
 - handling errors, 291–292
 - handling JSON data returned by the server, 290–291
 - interface, 286
- `range()` function, Python, 106–107, 121
- RapidAPI, 170
- `read()` method, Python, 146
- `readlines()` method, Python, 147
- `remove()` method
 - JavaScript DOMTokenList, 264, 266
 - Python, 109–110
- REPL (read, evaluate, print, loop), 72, 81, 85, 92, 93
- `replace()` method, Python, 100
- requests library, Python, 142–144, 166–167
- reserved words, 10
- `reverse()` method, JavaScript, 249
- `reversed()` function, Python, 107
- runtime errors, 45, 46
- Rust, 65–66

S

- Safari, 215, 272, 274
- sandbox, 52

- scientific computing, 20
- scikit-learn, 20
- script mode, Python, 81–82, 85–87
- `<script>` tag, 211, 213, 224, 225–226, 227, 254
- script values, monitoring, 46
- selectors, CSS, 308–312
 - definition of, 308
 - specifying DOM elements by, 255
- sentiment analysis, 202–204
- `sent_tokenize()` function, Python, 200
- server-side web development, 64
- `setDate()` method, JavaScript, 239
- `setFullYear()` method, JavaScript, 239
- `setHours()` method, JavaScript, 239
- `setMilliseconds()` method, JavaScript, 239
- `setMinutes()` method, JavaScript, 239
- `setMonth()` method, JavaScript, 239
- sets, Python, 114–115
- `setSeconds()` method, JavaScript, 239
- `setTime()` method, JavaScript, 239
- `shift()` method, JavaScript, 249
- single-line syntax, 34
- `slice()` method, JavaScript, 236, 249
- smart devices, 21
- `sort()` method, JavaScript, 249
- `sorted()` function, Python, 183, 198, 199
- `split()` method
 - JavaScript, 236
 - Python, 101–102
- `sqrt()` function, Python, 138, 139
- `src` attribute, 214
- `startsWith()` method, JavaScript, 235
- `startswith()` method, Python, 100
- statements, definition of, 11
- statistics module, Python, 135, 136, 137–138
- stepping (debugging), 45–46, 279
 - into a code, 280
 - one statement at a time, 279
 - out of a code, 281
 - over a code, 280–281
- stop words, 196–197
- `str()` function, Python, 93
- string expressions, 32
- string literals, 28, 90, 91, 232–234
- string module, Python, 196
- strings, 26
 - multiline, 233, 234
 - using variable values in, 233
- strings, JavaScript, 231–232
 - converting between strings and numbers, 240–242
 - extraction of substrings, 235, 236
 - length of, 234
 - searching for substrings, 235
 - templates, 232–234
- strings, Python, 91
 - getting input from user, 102–104
 - indexes, 97–98
 - looping through, 120
 - methods, 99–101, 100
 - mixing with variables, 99
 - splitting and joining, 101–102

- `strip()` method, Python, 147, 185
- style property of HTML elements, 264
- `substr()` method, JavaScript, 236
- `substring()` method, JavaScript, 236
- substrings, JavaScript, 235, 236
- Swift, 66–67
- synchronous operations, 288
- syntax
 - block, 34, 35
 - definition of, 10, 34
 - single-line, 34
- syntax errors, 44
- `sys.exit()` function, Python, 181, 195
- systems programming, 66

T

- template literals, 233–234
- TensorFlow, 20
- Terminal
 - launching on macOS, 74
 - launching on Windows, 73
 - shortcuts, 74–75
- text analyzer (Python project), 193
 - analyzing sentence length, 200–202
 - analyzing text sentiment, 202–204
 - cleaning the text, 195–196
 - finding longest words, 199
 - finding most common words, 196–199
 - full code of, 204–206
 - library installation for, 193–194
 - opening and reading text file, 195

- textContent property, JavaScript, 262–263
- thinking skills, and coding, 16
- time *see* date and time, JavaScript
- TIOBE Index, 50
- `title()` method, Python, 100
- `toggle()` method, JavaScript, 267
- traceback messages, Python, 170–171
- transistors, 11–12
- true/false decisions, 34–35
- try block, Python, 171–174, 195
- try/catch structure, JavaScript, 291
- tuples, Python, 111–112
- `type()` function, Python, 176
- type selector, 309
- Typescript, 67–68

U

- universal selector (*), 309
- unpacking, tuple, 112
- `unshift()` method, JavaScript, 249
- `upper()` method, Python, 100

V

- VADER (Valence Aware Dictionary and sEntiment Reasoner), 203
- var keyword
 - Go, 25
 - JavaScript, 220
- variable interpolation, 233
- variable scope, 227
 - JavaScript, 228–230
 - Python, 131–134

- variables, 24
 - arrays, 26
 - declaring, 24–25
 - inclusion in other statements, 25–26
- JavaScript, 25, 220–221
- lists, 26
- Python, 90, 99

vibe coding, 300

virtual reality (VR) applications, 59

VR *see* virtual reality applications

W

web application development, 63

web browsers, 15, 59, 210, 251

- debugging tools in, 214, 215, 272, 273
- HTML viewer in, 215, 216, 272, 273
- web development tools in,
214–218, 272–273
- see also* Console window in web browsers

web development, 20, 55, 56–57, 61, 63

- server-side, 64
- tools, in browsers, 214–218, 272–273
- see also* JavaScript

web server, 21, 55

WebAssembly, 66

while loop, 37

- JavaScript, 221, 223
- Python, 121–122, 128, 131,
184–185, 187, 188

while True statement, Python, 122, 151

Windows

- application development, 60
- checking Python installation status
in, 76, 77
- Console window in browsers of, 274
- installation of Python library on, 142
- installation of Python on, 77–79
- launching Terminal on, 73, 73
- running Python on, 82, 83
- running Python script file in, 86
- Terminal shortcuts, 74–75
- web development tools in browsers of,
214–217, 272

Windows PowerShell, 73

with statement, Python, 145–146

word_tokenize() function, Python, 200

write() method, Python, 145–146

Z

Zen of Python, 154