

INDEX

Symbols and Numbers

`==` (comparison) operator, 36–37
`%` (modulo) operator, 78, 255–256
`*` (splat) operator, 25
2of4brif.txt file, 20
4 Percent Rule, 239–240, 263

A

absolute paths, 329
abstractions, 21
acceleration, 286
actions, 223
actuarial life tables, 245
Adding Habitable Zones to Your
Galaxy project, 215–216
aerobraking, 321–322
Alexander, Edward Porter, 87
All the Marbles problem, 264
Anaconda, 195
anagrams, defined, 35
ancestors, 229, 272
apoapsis, 290
apogee, 290
application programming interfaces
(APIs), 267
artists, 360
assets, 296
attributes, 225
Audacity, 296
Automatic Anagram Generator
project, 62
Automating Possible Keys project, 88,
371–372

B

Babbage, Charles, 107
`bar_chart()` function, 355, 359, 358–360
Beating Benford project, 364–365,
387–388
Belyayev, Dmitry, 134
Benford, Frank, 347

Benfording the Battlegrounds
project, 366
Benford's law, 348–363
bigrams, 52
Birthday Paradox problem, 238
*The Black Swan: The Impact of the Highly
Improbable*, 2nd Edition
(Taleb), 263
black swans, 245
blitting and `blit()` method, 280, 317
block transfer, 280
block-level objects, 111
British brute-force code, 54–55
brute-force method, 40, 54–55
Building a Galactic Empire project,
213–214, 380–382
Building a Rat Harem project, 144

C

calculators, 240
candidates, 175
Canvas widget, 203
Carnegie Mellon University
Pronouncing Dictionary
(CMUdict), 148–153
case sensitive letters, 39
CCDs (charge-coupled devices), 325
CCDStack, 331
challenge projects
Adding Habitable Zones to Your
Galaxy, 215–216
All the Marbles, 264
Automatic Anagram Generator, 62
Benfording the Battlegrounds, 366
Building a Rat Harem, 144
With a Bullet, 284
Creating a Barred-Spiral Galaxy,
214–215
Creating a More Efficient
Safecracker, 144
The Fountainhead, 283
To Haiku, or Not to Haiku, 186

- Just My Luck! 264
 - Markov Music, 186
 - Mars Orbiter game modifications, 320–323
 - The Middle, 17
 - Mix and Match, 264
 - New Word Generator, 184
 - A Picture Is Worth a Thousand Dollars, 264
 - Poor Foreign Man’s Bar Chart, 16–17
 - Recursive Approach, 34
 - Route Cipher Encoder, 90
 - Shock Canopy, 283
 - Three-Rail Fence Cipher, 90
 - Turing Test, 185
 - Unbelievable! This Is Unbelievable! Unbelievable! 185–186
 - Using Monospace Font, 123
 - Vanishing Act, 344–345
 - While No One Was Looking, 366
 - charge-coupled devices (CCDs), 325
 - check_keys() method, 301
 - Checking the Number of Blank Lines project, 122–123, 377–378
 - chi-square goodness-of-fit test, 352–353
 - Churchill, Winston, 3
 - cipherlist, 78
 - ciphertext, 78
 - Clark, Brooks, 330
 - Class attributes, 269–270
 - classes, 223, 229, 298–299
 - clean_folder() function, 333
 - Clinton, Hillary, 353–354
 - CMUdict (Carnegie Mellon University Pronouncing Dictionary), 148–153
 - The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography* (Singh), 87, 101
 - coding conventions, 6
 - coding style, xxv
 - Colchester Catch project, 103, 376
 - collections module, 40
 - color tables, 297–298
 - command window, 7
 - columnspan, 231
 - comparison operator (==), 36–37
 - computer modeling, 194
 - computer performance evaluation, 162
 - consonant-vowel maps (c-v maps), 52–58
 - constant of proportionality, 307
 - continue statements, 178
 - control messages, 67
 - corpus, 148
 - count_first_digits() function, 356
 - Counter module, 40–41, 56–57, 197
 - counting syllables, 147–159
 - cProfile, 20, 30
 - Cracking Codes with Python* (Sweigart), 61, 87, 119
 - Creating a Barred-Spiral Galaxy project, 214–215
 - Creating a More Efficient Safecracker project, 144
 - crossover process, 128, 133
 - cryptography, 51
 - current working directory (cwd), 328
 - c-v maps (consonant-vowel maps), 52–58
 - cv_map_filter() function, 57
 - cv_words() function, 58
- ## D
- data
 - loading, 355–356
 - object-oriented programming (OOP) and, 223
 - data verification, 77–78
 - debugging, 169–171
 - Deep Sky Stacker, 331
 - defaultdict container type, 172–173, 355–357
 - default_input() function, 251–252
 - del_folders() function, 333
 - designed development, 2
 - Dictionary Cleanup project, 33, 368–369
 - dictionary files, 20–22, 160
 - digrams, 52–54, 72
 - directionality, 211
 - directory paths, 328–329
 - DirectX API*, 267
 - docstrings, 10–11
 - Downey, Allen, 2
 - Drake equation, 187, 190–191
 - draw() method, 280
 - duration variable, 254
 - DVDVideoSoft, 330
- ## E
- eccentricity, 301
 - eccentricity variable, 308–309, 314, 316
 - Effective Python* (Slatkin), 15

- electronic ink, 108
 - Elementary* (television show), 105
 - Emerson, Ralph Waldo, xxii
 - encapsulations, 21
 - enjambment, 167
 - enter key, 5
 - Enthought Canopy, 195
 - enumerate() function, 134
 - epoch timestamps, 31
 - errors. *See also* debugging
 - checking for, 253
 - false, 8
 - events, 278
 - evolutionary algorithms, 125
 - exceptions, 21
 - exhaustive search engines, 126
 - expected counts, 357
- F**
- Fermi's paradox, 187–211
 - files and folders, 327–329, 333–334
 - Finding Digrams project, 61, 369
 - first digits, 356–357
 - first-digit law. *See* Benford's law
 - Flake8, 7
 - flipping and flip() method, 280
 - font objects, 111
 - fonts
 - color, 108
 - types of, 109–110
 - using, 119
 - Fooled by Randomness: The Hidden Role of Chance in Life and in the Markets*, Revised Edition (Taleb), 263
 - for loops, 257
 - Fountainhead project, 283
 - 4 Percent Rule, 239–240, 263
 - Frame class, 229
 - frames per second (fps), 278–280
 - Frame widget, 229
 - Free Studio, 330
 - Free Video to JPG Converter tool, 330
 - Friedman, William, 65
 - functions, defining, 250, 309–310
- G**
- A Galaxy Far, Far Away project, 212–213, 379–380
 - game assets, 227
 - game loops, 299–300, 315–316
 - game sketches, 268–269, 293–295
 - Geany, xxiv
 - generations, 128–129
 - genetic algorithms, 125–142
 - Genetic Algorithms with Python* (Sheppard), 143
 - get_expected_counts() function, 357
 - getdata() method, 339
 - global minima and maxima, 140
 - global scope, 42
 - Going the Distance project, 282, 385–387
 - goodness-of-fit test, 352–353, 357–358
 - Google
 - PageRank algorithm, 162
 - style guide, 10–11, 14
 - graphical models, 199–201
 - graphical user interface (GUI), 74
 - gravity, 286–287, 307
 - groups, 314
 - Guns of the South* (Turtledove), 66
- H**
- Hacking Lincoln project, 87, 370
 - haiku, 146–147
 - To Haiku, or Not to Haiku project, 186
 - handwriting recognition, 162
 - Hartman, Charles, 145–146, 167, 182, 184
 - Highlighter tool, 108
 - hill-climbing algorithm, 139
 - Hitchhiker's Guide to Python*, 15
 - Hohmann transfer orbit, 291
 - human error, 66
- I**
- Identifying Cipher Types project, 88, 370–371
 - IDLE shell, 3
 - IDLE text editor, xxiv
 - image registration, 331
 - image stacking, 325
 - images, 334–337
 - import statements, 250
 - inline-level objects, 111
 - instantiation, 314
 - integrated development environment (IDE), xxiv
 - invisible electronic ink, 108
 - ipadx option, 231
 - itertools module, 51

J

Japanese Haiku (Beilenson), 146
JavaScript Object Notation (json), 152
Jumble, 36, 61
Just My Luck! problem, 264

K

Kepler, Johann, 287
kerning, 109–110
KEYUP event, 316
keys, 172

L

.lower() method, 6
Lanczos filter, 337
letter_pair_filter() function, 58–59
letter-transposition ciphers, 72
LibreOffice Writer, 107
line breaks, 110
list ciphers, 99–101
list comprehension, 22, 71
local minima and maxima, 140
logarithmic scale, 349
logarithmic spiral, 199
logging module, 170–171
logic, 223
loops
 game loops, 299–300, 315–316
 for loops, 257
 while loops, 5, 135, 254, 318
lower case, displaying characters as, 39

M

machine translations, 147
main() function, 42, 46–48, 60–61,
 76–79, 258–259, 277–278,
 278–280, 312–313, 361–363
Markov chain analysis, 147, 161–182
Markov Music project, 186
Mars Climate Orbiter, 6
Mars Orbiter game modifications
 project, 320–323
Mary (queen of Scotland), 91
matplotlib library, 195–196, 198, 248,
 250, 355
MCS (Monte Carlo simulation), 218–238
median, 344
methods, 223, 225
microphotography, 80

Microsoft

DirectX API, 267
Office Suite, 107
Outlook, 121
The Middle project, 15–16, 17
Milky Way, 188–189
Mix and Match problem, 264
model of order 0, 162
model of order 2, 162–163
modules, xxi, 21
 importing, 250, 355–356
 matplotlib, 195–196, 198, 248,
 250, 355
 pillow, 326–327
 pydocstyle, 11, 15
 pygame, 267, 281–282, 296–319
 Pylint, 7–13, 15
 python-docx, 110–111
 tkinter, 200–202, 212, 229–231
modulo operator (%), 78, 255–256
monospace fonts, 109–110
Monte Carlo simulation (MCS), 218–238
most_common() method, 56–57
*Mysterious Messages: A History of Codes
 and Ciphers* (Blackwood),
 87, 101
Myth Busters, 284

N

natural language processing (NLP), 147
*Natural Language Processing with Python:
 Analyzing Text with the Natural
 Language Toolkit* (Bird, Klein
 and Loper), 155–156
Natural Language Toolkit (NLTK),
 148–149
New Word Generator project, 184
Newcomb, Simon, 347
Newton, Isaac, 286–287
Nigrini, Mark, 364
null, defined, 92
null ciphers
 defined, 91–92
 writing, 98–101
NumPy docstring standards, 10
NumPy library, 195–196, 198

O

object-oriented programming (OOP),
 223, 267

- Ogg Vorbis format (.ogg), 296, 300
- O’Neill, Tip, 366
- One-Tangent Burn, 291
- OpenGL (Open Graphics Library), 267
- OpenOffice Writer, 107
- OpenSource Computer Vision (OpenCV), 343
- operating systems, xxiv
- optimization, 126, 139
- orbital mechanics, 288–292
- os.chdir() method, 328–329
- os.getcwd() method, 328–329
- os.listdir() method, 333
- os.path.abspath() method, 329
- os.path.isdir() method, 333
- os.path.join() method, 328–329
- os.path.normpath() method, 328–329
- os.remove() method, 333–334
- Oxford English Dictionary*, 2nd Edition, 51

P

- PageRank algorithm, 162
- palindromes, 19, 23–25
- palingrams, 19, 25–33
- paragraph objects, 111
- patches, 2
- path() method, 303
- pathnames, 328–329
- Peale, Stan, 265
- PEP 8, 6, 14
- PEP 257, 10–11, 14
- periapsis, 290
- perigee, 290
- permutations, 51
- permutations with repetition, 137
- persistent data, 155
- phonemes, 150
- phrase anagrams, 39
- phrase_anagrams.py*, 43–44
- A Picture Is Worth a Thousand Dollars problem, 264
- Pig Latin project, 15, 367–368
- pillow module, 326–327
- PIL (Python Imaging Library), 327
- pip (Preferred Installer Program), 7, 11, 110–111
- plaintext, 63, 68
- planetary motion, 287
- planning, importance of, 2
- polar coordinates, 200
- polynomial equations, 193

- Poor Foreign Man’s Bar Chart project, 16–17
- Poor Man’s Bar Chart project, 15–16, 368
- pop() function, 71, 86, 155
- PowerShell, 7
- practice projects
 - Automating Possible Keys, 88, 371–372
 - Beating Benford, 364–365, 387–388
 - The Birthday Paradox, 238
 - Building a Galactic Empire, 213–214, 380–382
 - Building a Rat Harem, 144
 - Checking the Number of Blank Lines, 122–123, 377–378
 - The Colchester Catch, 103, 376
 - Creating a More Efficient Safecracker, 144
 - Dictionary Cleanup, 33, 368–369
 - Finding Digrams, 61, 369
 - A Galaxy Far, Far Away, 212–213, 379–380
 - Going the Distance, 282, 385–387
 - Hacking Lincoln, 87, 370
 - Identifying Cipher Types, 88, 370–371
 - Pig Latin, 15, 367–368
 - Poor Man’s Bar Chart, 15–16, 368
 - A Roundabout Way to Predict Detectability, 214, 383–385
 - Route Transposition Cipher:
 - Brute-Force Attack, 88–89, 372–374
 - Saving Mary, 102–103, 375
 - Storing a Key as a Dictionary, 88, 371
 - Syllable Counter vs. Dictionary File, 160, 378–379
- predictive text, 147
- Preferred Installer Program (pip), 7, 11, 110–111
- prep_words() function, 55
- print() function, 3
- print statements, 5
- probability theory, 161. *See also* Markov chain analysis
- process_choice() function, 45–46
- product() iterator, 137–138
- profiles, 29–31
- prograde, 288–289

- project code, xxiv–xxv. *See also*
 - source code
 - Benford’s Law of Leading Digits, 355–363
 - Breeding an Army of Super-Rats, 130–136
 - Counting Syllables
 - Count Syllables, 156–158
 - Missing Words, 151–156
 - Cracking a High-Tech Safe, 140–142
 - Finding Palindromes, 24–25
 - Finding Palingrams, 28–29
 - Finding Phrase Anagrams, 43–49
 - Finding Single-Word Anagrams, 38–39
 - Finding Voldemort
 - British Brute-Force, 54–61
 - Gallic Gambit, 50
 - Generating Pseudonyms, 4–6
 - Hiding a Vigenère Cipher, 114–119
 - Markov Chain Analysis, 171–181
 - Mars Orbiter Game, 297–319
 - Modeling the Milky Way
 - Galaxy Simulator, 202–211
 - Probability-of-Detection, 194–199
 - Monty Hall Game, 228–238
 - Plumes of Io, 271–280
 - Rail Fence Cipher
 - Decryption, 84–86
 - Encryption, 82–84
 - Route Cipher Decryption, 69–79
 - Simulating Retirement Lifetimes, 250–259
 - Stacking Jupiter
 - Cropping and Scaling, 331–337
 - Enhancing, 340–342
 - Stacking, 337–340
 - Trevanion Cipher, 94–98
 - Verify vos Savant, 221–223
 - Writing a Null Cipher, 99–101
 - proportional fonts, 109–110
 - prototype and patch, 2
 - pseudocode
 - defined, 3–4
 - graphical example, 127–128
 - pseudonym generator, 1–13
 - pseudonyms.py*, 4–5
 - Psych*, 1
 - PyCharm, xxiv
 - pycodestyle, 7
 - pydocstyle, 11, 15
 - pygame, 267, 281–282, 296–319
 - PyLint, 7–13, 15
 - PyScripter, xxiv
 - Python Enhancement Proposals*, 6
 - Python IDLE text editor, xxiv
 - Python Imaging Library (PIL), 327
 - Python Standard Library, 170
 - python-docx, 110–111
- Q**
- quadratic equation, 193
 - QUIT event, 280
- R**
- radians, 275
 - rail fence cipher, 80–86
 - random module, 5, 134
 - raw string, 82
 - Recursive Approach project, 34
 - RegiStar, 331
 - RegiStax, 331
 - relative paths, 329
 - repeated random sampling, 218
 - reStructuredText*, 10–11, 15
 - retrograde, 288–289
 - rotate() method, 303, 305–306
 - Roundabout Way to Predict
 - Detectability project, 214, 383–385
 - Route Cipher Encoder project, 90
 - route transposition cipher, 64–79
 - Route Transposition Cipher: Brute-Force Attack project, 88–89, 372–374
 - run objects, 111
- S**
- Sagan, Carl, 211
 - Saving Mary project, 102–103, 375
 - scaffolding, 170
 - scale invariant, 348
 - SciPy, 195
 - SDL (Simple DirectMedia Library), 267
 - semilogarithmic plots, 349
 - semordnilap, 26
 - sensitivity studies, 136
 - serialization, 155–156
 - Shakespeare, William, 186
 - Shannon, Claude, 162, 184

SHARPEN filter, 342
 shell utilities module, 329
 Sheppard, Clinton, 143
 Shock Canopy project, 283
 shutil module, 329
 shutil.rmtree() method, 333
 Simple DirectMedia Library (SDL), 267
 Slatkin, Brett, 15
 slicing, 23–24
 sorted() function, 37
 sounds, 315–316
 source code. *See also* project code
 anagrams.py, 38
 benford.py, 355–361
 brute_force_cracker.py, 138
 count_syllables.py, 157–158
 crop_n_scale_images.py, 332–334
 elementary_ink.py, 114–116
 enhance_image.py, 340–341
 galaxy_simulator.py, 202–209
 load_dictionary.py, 22
 list_cipher.py, 99
 markov_haiku.py, 171–180
 mars_orbiter.py, 297–319
 missing_words_finder.py, 151–156
 monty_hall_gui.py, 229–237
 monty_hall_mcs.py, 221–222
 nest_egg_mcs.py, 250–258
 nest_egg_mcs_1st_5yrs.py, 261–262
 null_cipher_finder.py, 95–98
 palindromes.py, 24
 palingrams.py, 28–29
 palingrams_optimized.py, 32
 phrase_anagrams.py, 44–47
 probability_of_detection.py, 195–198
 pseudonyms.py, 4–5
 pseudonyms_main_fixed.py, 12–13
 Pylint, 7–13
 rail_fence_cipher_decrypt.py, 84–86
 rail_fence_cipher_encrypt.py, 82–84
 route_cipher_decrypt_prototype.py,
 69–72
 safe_cracker.py, 140–142
 stack_images.py, 338
 super_rats.py, 130–136
 test_count_syllables_w_full_corpus
 .py, 159
 tvashitar.py, 271–279
 voldemort_british.py, 54–61
 spam detection, 147
 spam filtering, 162
 speech recognition, 162
 Sphinx, 11
 spiral galaxies, 188–189
 spiral transfer, 292
 splat operator (*), 25
 split() function, 70
 Sprite class, 272, 274, 298, 304, 314
 Stager, Anson, 64
 standardized names and procedures, 6
 statistics, 217
 steganography, 91, 108
 stop conditions, 126
 Storing a Key as a Dictionary project,
 88, 371
 string format method, 210
 string.punctuation constant, 95–96
 strings, 5–6
 strip() function, 65
 style guide, 6
 styles, 111–112
 statistical models of, 168
 substitution cipher, 65, 88, 106
 superclasses, 229
 Syllable Counter vs. Dictionary File
 project, 160, 378–379
 syllables, counting, 147–159
 synchronous orbits, 292
 sys module, 5
 sys.exit(1), 21

T

tableau, 106
 Taleb, Nassim, 219, 263
 terminal window, 111
 text editors, 20
 “The Growing Importance of Natural
 Language Processing”
 (DeAngelis), 155–156
Think Python, 2nd Edition (Downey), 2,
 33, 61, 170
 third-party modules
 pillow, 326–327
 python-docx, 110–111
 resources for, 15
 Three-Rail Fence Cipher project, 90
 time.time(), 31
 Tk widget, 231
 tkinter, 200–202, 212, 229–231
 tracking, 109–110
 training corpus, 148, 151

- transposition cipher, 65
- Trevanion cipher, 91–98
- `trigram_filter()` function, 58
- trigrams, 52
- Trump, Donald, 185–186, 353–354
- try statement, 21
- Turing, Alan, 185
- Turing Test project, 185
- Turtledove, Harry, 66
- Twain, Mark, 239–240
- `2of4brif.txt` file, 20

U

- Unbelievable! This Is Unbelievable!
 - Unbelievable! project,
 - 185–186
- unbreakable cipher. *See* Vigenère cipher
- uncertainty, 245
- universal gravity, 286–287, 307
- Unix epoch, 31
- unpacking, 176
- `update()` method, 277, 303
- user interfaces, writing, 178–181
- Using Monospace Font project, 123

V

- `validate_col_row()` function, 78
- values, 172
- Vanishing Act project, 344–345

- variables, 136
 - duration variable, 254
 - eccentricity variable, 308–309, 314, 316
- video, 330–331
- Vigenère cipher, 106–121
- Virtual Muse: Experiments in Computer Poetry* (Hartman), 145, 167, 184
- vos Savant, Marilyn, 217–218

W

- while loops, 5, 135, 254, 318
- While No One Was Looking
 - project, 366
- widget, defined, 203
- With a Bullet project, 284
- with statement, 21
- word association norms (WANs), 184
- word lists, 20

Y

- Yahoo! Mail, 121

Z

- Zatara, Zatanna, 19, 33
- Zen of Python*, 6
- `zip()` function, 133, 141
- `zip_longest()` function, 85–86
- `zorder` attribute, 360