

# INDEX

## Symbols

+ (addition and concatenation operator), 12, 23–24, 89–90  
+= (augmented addition assignment operator), 91–92  
/= (augmented division assignment operator), 92  
\*= (augmented multiplication assignment operator), 92  
-= (augmented subtraction assignment operator), 92  
\  
(backslash), 28–29  
:  
(colon), 26  
/  
(division operator), 12  
"  
(double quote), 29–30  
\_\_  
(double underscore), 95  
==  
(equal to comparison operator), 43–45  
\*\*  
(exponent operator), 201  
>  
(greater than comparison operator), 43–44  
>=  
(greater than or equal to comparison operator), 43–44  
#  
(hash mark), 34  
//  
(integer division operator), 181  
<  
(less than comparison operator), 43–44  
<=  
(less than or equal to comparison operator), 43–44  
%  
(modulo operator), 173  
\*  
(multiplication and replication operator), 12, 24, 89–90  
\  
(newline escape character), 28–29  
!=  
(not equal to comparison operator), 43–45  
()  
(parentheses), 14, 28, 35  
'  
(single quote), 22, 28–30  
[]  
(square brackets), 26  
%s  
(string formatting), 75

-  
(subtraction operator), 12  
\  
(tab escape character), 28–29  
''' or ''''  
(triple quotes), 164–165

## A

absolute path, 131  
addition operator (+), 12  
Adleman, Leonard, 337  
affine cipher, 177–181  
    decrypting with, 179–180, 194–195  
    encrypting with, 179, 193–194  
    hacking, 201–204  
*affineCipher.py*, 186–187  
*affineHacker.py*, 198–199  
*affineKeyTest.py*, 192  
*al\_sweigart\_privkey.txt*, 342, 343  
*al\_sweigart\_pubkey.txt*, 342, 343, 346, 363  
and operator, 106, 108  
append() list method, 121, 155–156, 253–254  
arguments, 28  
    default, 157  
    defining functions with  
        parameters, 83–84  
    for file permissions, 131, 132  
    keyword  
        end, for print(), 306  
        key, for sort(), 273  
        repeat, for itertools.product(), 307  
        reverse, for sort(), 273, 275  
    lists as, 121–122, 301  
    optional, 131  
    passing references as, 121, 234  
assignment statements, 16  
augmented assignment operators, 91–92  
authentication, 337–338

## B

- Babbage, Charles, 247, 282
- backslash (\), 28–29
- Bellaso, Giovan Battista, 247
- Bernstein, Daniel J., xxi–xxii
- blocks
  - of code, 45–46
  - in cryptography, 350
    - converting block to string, 354–355
    - converting string to block, 350–352
- Boolean operators, 106–108
- Boolean values, 43
- breakpoints, for debugging, 379–380
- break statement, 311–312
- brute-force attack, 69
- built-in functions, 56

## C

- Caesar cipher, 4
  - decrypting with, 7
  - encrypting with, 7
  - hacking, 69
- caesarCipher.py*, 54–55
- caesarHacker.py*, 70–71
- candidates (possible cipherwords), 223
- case (of letters)
  - converting with string methods, 134, 216–217
  - sensitivity, in Python, 18
- `ceil()` function (*math* module), 103–104
- chiffre indéchiffrable, 247
- cipher disk, 4
- cipherletters, 222
- cipherletter mappings, 224, 232–241
  - adding letters to, 233–234
  - intersecting, 234–235
- ciphers, vs. codes, 3–4
- ciphertext, 4
- cipher wheel, 4
  - decrypting with, 6–7
  - encrypting with, 5–6
- cipherwords, 222
- clock arithmetic, 172–173
- `close()` file object method, 132
- code breaker, 2
- codes, vs. ciphers, 3–4
- colon (:), 26
- comments, 34
- comparison operators, 43–45

- composite numbers, 323, 326–327
- concatenation operator (+), 23–24, 89–90
- condition, 42–43
- confidentiality, 338
- constants, 57, 313
- continue statement, 202–204
- `copy.deepcopy()` function, 122
- `copy()` function (*pyperclip* module), 56, 65
- cryptanalyst, 2
- cryptographers, 2
  - Adleman, Leonard, 337
  - Bellaso, Giovan Battista, 247
  - Kerckhoffs, Auguste, 70
  - Rivest, Ron, 337
  - Schneier, Bruce, 337
  - Shamir, Adi, 337
  - Turing, Alan, 188, 268, 352
  - Vigenère, Blaise de, 247
- cryptography, 2
- cryptomath.py*, 182
- Cuisenaire rods, 174–175

## D

- data types, 13
  - Boolean, 43
  - dictionary, 146–150
  - floating-point, 13
  - integer, 13
  - list, 86–90
  - string, 22
  - tuple, 190
- debugging
  - in IDLE, 375–379
  - setting breakpoints, 379–380
- decoding, 3
- decrypting, 2
  - with the affine cipher, 179–180, 194–195
  - with the Caesar cipher, 7
  - with a cipher wheel, 6–7
  - with the public key cipher, 368
  - with the reverse cipher, 39
  - with the simple substitution cipher, 216
  - with the transposition cipher, 100–101
  - with the Vigenère cipher, 248–249
- `deepcopy()` function (*copy* module), 122
- `def` statement, 82–84
- detectEnglish.py*, 143–144

- dictionaries, 146–150
  - adding or changing items in, 147
  - in operator and, 148
  - len() function and, 148
  - lists, differences from, 147
- dictionary attacks, 250–251, 280
- dictionary.txt*, 280
- diff tool, checking source code with, 31–32
- digital signatures, 338
- division operator (/), 12
  - integer (//), 181
- divisors. *See* factors
- double encryption, 8
- double quote ("), 29–30
- dunder (\_\_ , double underscore), 95
- dunder name dunder (\_\_name\_\_)
  - variable, 95–96
- duplicating lists, 122

**E**

- elif statement, 61
- else statement, 60
- encoding, 3
- encrypting
  - with the affine cipher, 179, 193–194
  - with the Caesar cipher, 7
  - with a cipher wheel, 5–6
  - with the public key cipher, 367–368
  - with the reverse cipher, 39
  - with the simple substitution cipher, 215
  - with the tranposition cipher, 78–80
  - with the Vigenère cipher, 248–249
- encryption key, 4
- end keyword argument, for print(), 306
- endswith() string method, 135
- English, detecting programmatically, 141–143, 156–159
- equal to operator (==), 43–45
- Eratosthenes, sieve of, 328–331
- errors, 15. *See also* debugging
  - finding with diff tool, 31–32
  - ImportError, 55
  - IndexError, 25
  - ModuleNotFoundError, 226
  - NameError, 33
  - SyntaxError, 15, 28, 29
- escape characters, 28–30
- ETAOIN, 260
- Euclid’s algorithm, 176–177
  - extended, 181

- execution, program
  - defined, 34
  - terminating with sys.exit(), 124
- exists() function (os.path module), 133
- exit() function (sys module), 124
- exiting programs, 35–36, 124
- exponent operator (\*\*), 201
- expressions, 12–15
  - defined, 13
  - evaluating, 14–15
- extend() list method, 301

**F**

- factors
  - of composite numbers, 323
  - greatest common divisors, 173–175, 176–177
  - of prime numbers, 322, 357
  - of spacings, in Kasiski examination, 283–284, 297
- False (Boolean value), 43
- file editor (IDLE), 30
- file objects
  - close() file object method, 132
  - open() function, 131
  - read() file object method, 132
  - write() file object method, 131
- files, plain text, 128, 130–134
- find() string method, 62–63
- float, 13
- float() function, 154
- floating-point numbers, 13
- floor() function (math module), 103–104
- for statement, 58–59
- freqAnalysis.py*, 265–266
- frequency
  - defined, 260
  - of English letters, 261
- frequency analysis, 259, 285–287
- frequency match score, 262–264, 286
- functions, 28
  - calling from modules, 65
  - passing as values, 272–273

**G**

- Gadsby* (Wright), 266
- gcd() function, 176–177
- GCD (greatest common divisor), 173–175, 176–177. *See also* factors

- generating keys
  - affine cipher, 195–196
  - public key cipher, 340
  - simple substitution cipher, 218
- global scope, 84
- googol, 322
- greater than operator (>), 43–44
- greater than or equal to (>=) operator, 43–44
- greatest common divisor (GCD), 173–175, 176–177. *See also* factors

## H

- hacker, 2
- hacking
  - affine cipher, 201–204
  - Caesar cipher, 69
  - public key cipher, why we can't, 355–357
  - simple substitution cipher, 222–225, 241–245
  - transposition cipher, 166–168
  - Vigenère cipher, 280–282
- hash mark (#), 34
- hello.py*, 31
- hybrid cryptosystems, 347

## I

- IDLE, xxvii, 12, 30
  - debugging with, 375–379
  - opening programs, 34
  - running programs, 33
  - saving programs, 32
- if statement, 59–60
- ImportError, 55
- import statement, 55, 56
- IndexError, 25
- indexing, 24–27
- infinite loop, 195
- in operator, 61–62
- in place modification, of lists, 122
- input() function, 35, 50
- insert() list method, 367
- installing Python, xxv–xxvi
- integer division operator (//), 181
- integer (int) data type, 13
- interactive shell, xxvii, 12
- intersected mapping, 225, 234–235
- int() function, 154

- islower() string method, 216–217
- isupper() string method, 216–217
- itertools module, 307–308

## J

- join() string method, 93–94, 253–254

## K

- Kasiski examination, 282–284
- Kasiski, Friedrich, 282
- Kerckhoffs, Auguste, 70
- Kerckhoffs's principle, 70
- key, encryption, 4
- key keyword argument, for sort(), 273

## L

- len() function, 41–42
  - using with dictionaries, 148
  - using with lists, 89
- less than operator (<), 43–44
- less than or equal to operator (<=), 43–44
- list-append-join process, 253–254
- list() function, 123, 213, 253–254, 274–275
- lists, 86–87
  - append() method, 121, 155–156, 253–254
  - as arguments, 121–122, 301
  - concatenating, 89–90
  - dictionaries, differences from, 147, 149
  - duplicating, 122
  - extend() method, 301
  - in operator, 89
  - insert() method, 367
  - len() method, 89
  - in place modification, 122
  - references, 119–121
  - replicating, 89–90
  - sort() method, 212–213, 271–275
- local scope, 84
- loops
  - for, 58–59
  - with the range() function, 72–73
  - while, 42–43
- Lovelace, Ada, 163
- lower() string method, 134

## M

machine-in-the-middle (MITM)  
    attack, 339  
'\_\_main\_\_' string value, 95  
main() function  
    in *affineCipher.py*, 188–189, 196  
    in *affineHacker.py*, 200, 204  
    in *makePublicPrivateKeys.py*, 343  
    and `__name__` variable, 95  
    in *publicKeyCipher.py*, 373  
    in *simpleSubCipher.py*, 211, 219  
    in *simpleSubHacker.py*, 231, 245  
    in *transpositionDecrypt.py*,  
        102–103, 110  
    in *transpositionEncrypt.py*, 85, 95  
    in *transpositionFileCipher.py*,  
        132–133, 138  
    in *transpositionHacker.py*,  
        164–165, 169  
    in *transpositionTest.py*, 118, 124  
    in *vigenereCipher.py*, 252–253, 257  
    in *vigenereHacker.py*, 294, 313  
*makePublicPrivateKeys.py*, 340–341  
*makeWordPatterns.py*, 225  
man-in-the-middle (MITM) attack, 339  
mappings. *See* cipherletter mappings  
math.ceil() function, 103–104  
math.floor() function, 103–104  
math operators  
    + (addition operator), 12  
    / (division operator), 12  
    \*\* (exponent operator), 201  
    % (modulo operator), 173  
    \* (multiplication operator), 12  
    - (subtraction operator), 12  
math.sqrt() function, 327, 356  
max() function, 364  
methods  
    defined, 62  
    passing as values, 272–273  
min() function, 364  
MITM (man-in-the-middle or machine-  
    in-the-middle) attack, 339  
modular arithmetic, 172–173  
modular inverse, 180–181  
    in affine cipher, 179–180, 181  
    in public key cipher, 344  
ModuleNotFoundError, 226  
modules  
    calling functions from, 65  
    importing, 56  
modulo operator (%), 173

Morse code, 3  
Morse, Samuel, 3  
multiline strings, 164–165  
multiple assignment, 175–176  
multiplication operator (\*), 12  
multiplicative cipher, 177–181

## N

`__name__` variable, 95–96  
NameError, 33  
negative indexes, 25–26  
newline  
    escape character (`\n`), 28–29  
    removing with the `strip()`  
        method, 167  
not equal to (`!=`) operator, 43–45  
not in operator, 61–62  
not operator, 107

## O

one-time pad cipher, 316  
open() function, 131  
opening programs, 34  
operators, 12  
    assignment, 16  
    augmented assignment, 91–92  
    comparison, 43–45  
    in, 61–62  
    math, 12  
    not in, 61–62  
order of operations, 14  
or operator, 107, 108  
os.path.exists() function, 133–134

## P

parameters. *See also* arguments  
    defining default arguments, 157  
    defining functions with, 83–84  
    optional, 131  
    passing references as, 121, 234  
    scope of, 84  
parentheses (`()`), 14, 28, 35  
*passingReference.py*, 121  
path (of files), 131  
pattern object, regular expression, 231  
permutation, 78  
PKI (public key infrastructure), 338  
plaintext, 4  
plain text files, 128, 130–134  
potential decryption letters, 223

- pow() function, 353–354
- precedence, 14
- primality test, 322
- prime factorization, 323
- prime numbers, 322–323
  - finding
    - with sieve of Eratosthenes, 328–330
    - with trial division algorithm, 326–327
  - use in public key cryptography, 340, 356–357
- primeNum.py*, 324–326
- print() function, 27–28, 34–35, 306
- product() function (itertools module), 307–308
- pseudorandom numbers, 116–117
- public key cipher
  - decrypting with, 368
  - encrypting with, 367–368
  - hacking, why we can't, 355–357
- publicKeyCipher.py*, 357–360
- public key cryptography, 336
  - example key files, 342
  - formula to generate keys, 340
- public key infrastructure (PKI), 338
- pyperclip.copy()* function, 56
- pyperclip.py*, xxvi

## Q

- quotes
  - enclosing strings
    - multiline, 164–165
    - single line, 29–30
  - escaping in strings, 28–29

## R

- Rabin-Miller primality test, 331–333
- random numbers, 116–117
- random.randint() function, 116–117
- random.seed() function, 116–117
- random.shuffle() function, 122
- range() function, 72–73
- read() file object method, 132
- references, to lists, 119–121
- regex objects, 231
- regular expressions
  - calling sub() method on, 241
  - finding characters with, 230–231
- repeat keyword argument, for
  - itertools.product(), 307

- replace() string method, 243
- replication, 24, 118
- replication operator (\*), 24, 89–90
- return statement, 94–95
- return value, 35, 94–95
- reverse cipher, 39
- reverseCipher.py*, 40
- reverse keyword argument, for sort(), 273, 275
- Rivest, Ron, 337
- round() function, 103–104
- RSA cipher, 337
- running programs, 33
- Russell, Bertrand, 209

## S

- saving programs, 32
- Schneier, Bruce, 337
- scope, 84
- secrets module, 318
- seed() function (random module), 116–117
- set() function, 298
- Shamir, Adi, 337
- Shannon, Claude, 70
- Shannon's Maxim, 70, 190
- shuffle() function (random module), 122
- sieve of Eratosthenes, 328–331
- signatures, digital, 338
- silent mode, 306
- simpleSubCipher.py*, 209–210
- simpleSubHacker.py*, 226–229
- simple substitution cipher, 208
  - decrypting, 216
  - encrypting, 215
  - hacking, 222–225, 241–245
- single quote ('), 22, 28–30
- slicing, 26–27
- sort() list method, 212–213, 271–275
- source code, 31
- spaces, removing with the strip()
  - method, 167
- split() string method, 150–151
- sqrt() function (math module), 327, 356
- square brackets ([]), 26
- square root, 327
- startswith() string method, 135
- statements, 16
  - assignment, 16
  - break, 311–312
  - continue, 202–204

- def, 82–84
- elif, 61
- else, 60
- for, 58–59
- if, 59–60
- import, 55, 56
- return, 94–95
- while, 42–43
- str() function, 154
- strings, 22–27
  - building with list-append-join process, 253–254
  - concatenation, 23
  - endswith() method, 135
  - escape characters, 28–30
  - find() method, 62–63
  - formatting (%s), 75
  - indexing, 24–27
  - interpolation, 75
  - islower() method, 216–217
  - isupper() method, 216–217
  - join() method, 93–94, 253–254
  - lower() method, 134
  - multiline, 164–165
  - quotes, 28–30
  - replace() method, 243
  - replication, 24, 118
  - slicing, 26–27
  - split() method, 150–151
  - startswith() method, 135
  - str() function, 154
  - strip() method, 167
  - title() method, 134–135
  - upper() method, 134
- stringTest.py*, 254
- strip() method, 167
- sub() regular expression method, 241
- subtraction operator (-), 12
- symbol set, 57
  - locating symbols in, 63
  - wraparound in, 64, 177–178, 193
- SyntaxError, 15, 28, 29
- sys.exit() function, 124
- sys.path.append() function, 226

**T**

- tab
  - escape character (\t), 28–29
  - removing with the strip() method, 167
- terminating programs, 35–36

- testing programs, 113–114
- textbook RSA, 337
- time.time() function, 136–137
- title() string method, 134–135
- transposition cipher, 77, 78–81
  - decrypting with, 100–101
  - encrypting with, 78–80
  - hacking, 166–168
- transpositionDecrypt.py*, 101–102
- transpositionEncrypt.py*, 81–82
- transpositionFileCipher.py*, 130
- transpositionHacker.py*, 162–163
- transpositionTest.py*, 114–115
- trial division algorithm, 326–327, 328
- triple quotes (''' or '''), 164–165
- True (Boolean value), 43
- tuples, 190, 268
- Turing, Alan, 188, 268, 352
- two-time pad cipher, 319–320

## U

- upper() string method, 134

## V

- Vail, Alfred, 3
- values, 12
- variables, 15–18
  - names, 18
  - overwriting, 17
- Vigenère, Blaise de, 247
- Vigenère cipher, 248–251, 316, 319–320
  - decrypting with, 248–249
  - encrypting with, 248–249
  - hacking, 280–282
- vigenereCipher.py*, 251–252
- vigenereDictionaryHacker.py*, 280–281
- vigenereHacker.py*, 251–252

## W

- while statements, 42–43
- whitespace characters, 167
- word patterns, 222–223
- wordPatterns.py*, 225
- wraparound
  - in modular arithmetic, 172–173
  - of symbol sets, 64, 177–178, 193
- wrapper functions, 213–214
- Wright, Ernest Vincent, 266
- write() file object method, 131