INDEX

Symbols

!= (inequality operator), 29
' (single quotes), 6
" (double quotes), 6
""" (triple-quoted strings), 150
() (parentheses), 11
* (asterisk operator), 7, 9, 104
** (exponentiation operator), 20
+ (plus sign operator), 7, 9, 104
- (minus sign operator), 9
. (dot operator), 8
/ (floating-point division operator), 10
// (integer division operator), 10
< (less-than operator), 28
<= (less-than-or-equal-to operator), 28
== (equality operator), 29
> (greater-than operator), 28
>= (greater-than-or-equal-to operator), 28
[] (square brackets), 88, 103, 109, 220
% (mod operator), 10, 82–84
`\n` sequence, 175
{} (curly brackets), 85, 210, 211, 218

A

Action Figures problem, 153
    boxes, representation, 154
    challenge, 153
    input, 154
    output, 154
    top-down design, 154
        check boxes, 158
        doing, 155
        new boxes, 160
        organize boxes, 163
        read input, 157
        sort boxes, 162
        top level, 155
`add` method, 212, 215
Algorithmic Thinking (Zingaro), 264
algorithms
    brute-force, 237
    complete-search, 237, 238
    constant-time, 268, 269
    cubic-time, 277
    defined, 237
    linear-time, 270, 271, 274
    log time, 279
    n log *n* time, 280
    quadratic-time, 274, 275
aliases, 107
`append` method, 111, 115, 139
arguments, 8
assignment statement, 11, 111
Automate the Boring Stuff with Python, 2nd edition (Sweigart), 201

B

Baker Bonus problem, 128
    challenge, 128
    input, 128
    output, 129
    problem solving, 132
    table, 129
        nested lists, 130
big O, 268–284, 295
    constant time, 268
    cubic time, 276
    function call, 282
    linear time, 269
    log time, 279
    multiple variables, 277
    n log *n* time, 280
    quadratic time, 273
        alternate form, 275
        typical form, 274
binary search, 256, 279, 280
bisect module, 258–260
`bisect_left`, 258, 259, 261, 262
`bisect_right`, 258, 259, 261, 262
block, 31
Boolean expression, `if` statement, 31
Boolean expression, `while` loop, 73
Boolean operators, 37, 38, 78

  - and, 39
  - not, 40
  - or, 38
- Boolean type, 27, 28
- Boolean values, 28
- break, 96
- brute-force algorithms. *See* complete-search algorithms
- bug, 55

C

- *C Programming: A Modern Approach*, 2nd edition (King), 296
- calling a method, 8
- Card Game problem, 137, 169
  - challenge, 138
  - defining and calling functions, 140
    - functions with arguments, 141
    - functions without arguments, 141
    - keyword arguments, 143
    - local variables, 144
    - mutable parameters, 145
    - return values, 147
  - function documentation, 149
  - high card, 138
  - input, 138
  - output, 138
  - problem solving, 150
  - rules, 138
  - types of cards, 138
- `cd` command, 4
- `center` method, 109
- `char`, 50
- `choice` function, 258
- Cities and States problem, 231
  - challenge, 231
  - input, 231
  - output, 231
  - problem solving, 234
- cleaning an email address, 205
- closing files, 179
- code duplication, avoiding, 116, 137
  - huge size, 116
  - list of sizes, building, 117
- comment lines, 43
- comments, 42, 43
- Common Words problem, 215
  - challenge, 215
  - dictionaries, 218
  - indexing dictionaries, 220
  - input, 216
  - inverting dictionaries, 225
  - looping through dictionaries, 222
  - output, 216
  - problem solving, 227
    - code, 227
    - kth most common words, 229
    - main program, 230
    - suffix, adding, 229
- complete-search algorithms, 237–239, 245
  - Cow Baseball problem, 249
  - Lifeguards problem, 238
  - Ski Hills problem, 244
- *Computers and the World of the Future* (Greenberger), 201
- conditional execution, 27
- Cone Volume problem, 1, 18
  - challenge, 19
  - input, 19
  - math in Python, 19
    - exponents, 19
    - pi, 19
  - output, 19
  - problem solving, 22
  - strings and integers, converting, 20
- constant-time algorithms, 268, 269
- continue, 96, 98
- `count` method, 8, 9, 109
- Cow Baseball problem, 249
  - bisect module, 258
  - challenge, 249
  - input, 249
  - nested loops, 250
    - code, 250
    - program efficiency, 252
  - output, 250
  - problem solving, 261
  - Python modules, 256
  - sorting first, 253
    - code, 253
    - program efficiency, 255
- cubic-time algorithm, 277
- curly brackets, 85, 210, 211, 218

D

- Data Plan problem, 63, 102
  - challenge, 63

  - input, 63
  - loops, 64
  - output, 64
  - problem solving, 64
- debugging, 55
- decision making. *See also* `if` statements
  - Telemarketers problem, 37
  - Winning Team problem, 25
- `def` keyword, 141
- defining and calling functions, 140
- definite loops, 47, 51. *See also* `for` loops
  - Data Plan problem, 63
  - Occupied Spaces problem, 56
  - Three Cups problem, 47
- dict, 219
- dictionaries, 203, 218–220
  - `get` method, 221
  - indexing, 220
  - inverting, 225
  - `items` method, 224
  - keys, 218
  - `keys` method, 223
  - looping through, 222
  - values, 218
  - `values` method, 223
- `dir` function, 108, 109
- division operators, 10
- DMOJ judge, xxvi, 18
- docstring, 149, 150, 161, 164
- documentation string, 149
- dot operator (.), 8, 11
- double quotes, 6
- double-counting, 233, 235

E

- *Effective Python*, 2nd edition (Slatkin), 236
- efficiency class, 268
- elements, 103
- `elif`, 33, 49
- `else`, 33
- Email Addresses problem, 204
  - challenge, 204
  - cleaning steps, 205
  - input, 204
  - list, efficiency of searching, 208
  - list, using a, 205
    - cleaning email address, 205
    - main program, 207
  - output, 205
  - problem solving, 214
  - sets, 210
    - efficiency of searching, 213
    - methods, 212
- escape character, 175
- Essay Formatting problem, 172
  - challenge, 172
  - input, 172
  - output, 172
  - problem solving, 179
    - code, 180
  - working with files, 172
    - closing files, 179
    - opening files, 173
    - reading from files, 174
    - writing to files, 177
- execution time, 267
- exponential function, 280
- exponentiation operator (**), 20
- exponents, 19
- expression, 9
- `extend` method, 112

F

- f-strings, 84–86, 139
- Farm Seeding problem, 182
  - challenge, 182
  - input, 183
  - output, 183
  - top-down design, 187
    - cow identification, 190
    - grass type elimination, 191
    - read input, 188
    - smallest-numbered grass type, 193
    - top level, 187
    - write output, 195
- files, 172
  - `close` method, 179
  - closing, 179
  - `open` function, 173
  - opening, 173
  - reading from, 174
  - `readline` method, 174, 175
  - `write` method, 177–179
  - writing to, 177
- `find` method, 206
- `float` function, 121

- floating-point numbers, 9–14, 20
- `for` loops, 49–52, 56, 57, 69, 72, 87, 176, 226, 270
  - block, 51
  - if statement inside, 51
  - iteration, 65
  - limitations, 72, 92
  - range, 72
  - variables, 49
- `for` statement, 49
- functions, 15, 137, 140
  - Action Figures problem, 153
  - Card Game problem, 137
  - `choice`, 258
  - define and create, 141
  - `dir`, 108, 109
  - documentation, 149–150
  - exponential, 280
  - `float`, 121
  - `help`, 109
  - `input`, 15, 140, 171, 174
  - `int`, 21
  - `invert_dictionary`, 226
  - keyword arguments, 143
  - `len`, 50, 211, 219
  - `list`, 61
  - local variables, 144
  - logarithm, 280
  - `max`, 117
  - `min`, 117
  - mutable parameters, 145
  - nonmodule, 258
  - `open`, 173
  - parameter, 142
  - `print`, 16, 31, 32, 44, 51, 85, 140, 171
  - `randint`, 257, 258
  - `range`, 59
  - return values, 147
  - `search`, 213, 214
  - `str`, 22
  - `sum`, 126, 130
  - `type`, 27
  - with arguments, 141
  - without arguments, 141

G

- `get` method, 221
- Gmail address, 204. *See also* Email Addresses problem
- grid variables, 133

H

- `help` function, 109
- high card, 138, 139

I

- `if` statements, 25, 27, 30, 31, 49, 78, 141
  - block of, 31, 32
  - `for` loops, inside, 51
  - `if` by itself, 31
  - `if` with `elif`, 32
  - `if` with else, 33
  - `isupper` in, 52
  - logic, 43, 49
- `in` operator, 96, 208, 213, 215, 260
- indefinite loops, 51. *See also* `while` loops
  - Secret Sentence problem, 91
  - Slot Machines problem, 69
  - Song Playlist problem, 86
- index error, 89
- `index` method, 126, 260
- indexing, 57–59, 89, 93, 105, 107
  - dictionaries, 220
  - range `for` loops, 61
- infinite loops, 74
- `input` function, 15, 140, 171, 174
- input redirection, 44
- input-process-output model, 2
- `insert` method, 112
- `int` function, 21
- integer interpretation, 183
- integers, 9–14
- interactive programs, 2
- `intersection` method, 213
- `invert_dictionary` function, 226
- `isupper` method, 52, 109
- `items` method, 224
- iteration of loops, 50

J

- `join` method, 120, 121
- judge, 18. *See also* programming judges

K

- keys, 218
- `keys` method, 223
- keyword, 31
- keyword arguments, 143, 144

L

- `len` function, 50, 211, 219
- Lifeguards problem, 238
  - challenge, 238
  - input, 238
  - output, 239
  - problem solving, 240
    - lifeguard, firing one, 240
    - main program, 241
    - program efficiency, 242
- linear-time algorithms, 270, 271, 274
- linear-time loop, 281
- `list` function, 61
- lists, 101
  - * operator, 104
  - + operator, 104
  - `append` method, 111, 115, 139
  - Baker Bonus problem, 128
  - `extend` method, 112
  - `index` method, 126, 260
  - indexing, 105
  - `insert` method, 112
  - methods, 110
  - mutability, 106
  - parallel, 124
  - `pop` method, 112
  - positions, 115
  - `remove` method, 113
  - School Trip problem, 118
  - slicing, 105
  - `sort` method, 111, 112, 162, 291
  - sorting, 110
  - square brackets to delimit, 103
  - summing, 126
  - values, 103
  - Village Neighborhood problem, 101
- local variables, 144
- logarithm function, 280
- logarithmic-time algorithm, 280
- logic error, 55
- Longest Scarf problem, 284
  - algorithm 1, 285
  - algorithm 2, 286
  - challenge, 284
  - input, 284
  - output, 285
- loop variables, 49, 62
- looping through keys, 223
- loops. *See* definite loops; indefinite loops
- `ls` command, 4

M

- mathematical operators, 9. *See also* operators
- `max` function, 117
- methods, 8, 108, 110, 212
- `min` function, 117
- mod operator (%), 10, 82–84
- mutability, 106, 107

N

- `NameError`, 87
- negative indices, 58
- nested lists, 130
- nesting, 51
  - levels, 81
  - loops, 77
- nondecreasing, 159
- nonmodule function, 258

O

- object-oriented programming (OOP), 169
- Occupied Spaces problem, 56
  - challenge, 56
  - indexing, 57
  - input, 56
  - loops, 56
    - range `for`, 59
    - range `for`, indices through, 61
  - output, 56
  - problem solving, 62
- opening files, 173
- operators
  - !=, 29
  - *, 7, 9, 104
  - **, 20
  - +, 7, 9, 104
  - -, 9
  - ., 8
  - /, 10

  //, 10
  <, 28
  <=, 28
  ==, 29
  >, 28
  >=, 28
  %, 10, 82–84
  Boolean, 37, 38, 78
  `in`, 96, 104, 208, 213, 215, 260
  linear time, 271
  relational, 78

P

parallel lists, 124
parentheses, 11
pi (π), 19
`pop` method, 112
`print`
  calls, 50, 132, 153, 182
  function, 16, 31, 32, 44, 51, 85, 140, 171
problem instance, 2
problem solving, xxii
  Action Figures problem, 165
  Baker Bonus problem, 132
  Card Game problem, 150
  Cities and States problem, 234
  Common Words problem, 227
  Cone Volume problem, 22
  Cow Baseball problem, 261
  Data Plan problems, 64
  Email Addresses problem, 214
  Essay Formatting problem, 179
  Farm Seeding problem, 196
  Lifeguards problem, 240
  Longest Scarf problem, 286
  Occupied Spaces problem, 62
  Ribbon Painting problem, 290
  School Trip problem, 127
  Secret Sentence problem, 95
  Ski Hills problem, 246
  Slot Machines problem, 79
  Song Playlist problem, 90
  Telemarketers problem, 41
  Three Cups problem, 54
  Village Neighborhood problem, 114
  Winning Team problem, 35
  Word Count problem, 16
problem with timing, 266
  execution time, 267
    computer, depends on, 266
    test case, depends on, 267
  program implementation, 267
  slowness, 267
programming
  defined, 1
  folder, 3, 4
programming judges, xxiv
  DMOJ judge, xxvi
  Timus judge, xxvi
  USACO judge, xxvi
programming language, xxii
Python
  features, 3
  installation, xxiii
    Linux, xxiii
    macOS, xxiii
    Windows, xxiii
  learning, xxii
  teaching language, xxiii
Python 2, xxiii
Python 3.0, xxiii
Python 3.6, 86
Python Algorithms, 2nd edition (Hetland), 264
Python Crash Course, 2nd edition (Matthes), 169, 236
Python modules, 256–258
Python shell, 2–5, 17, 209, 214
  defined, 3
  launch, 3
  Linux, 4
  macOS, 4
  typing expressions at, 15, 16
  Windows, 3
Python types
  dictionaries, 218. *See also* dictionaries
  floating-point numbers, 9, 10
  integers, 9
  lists, 103. *See also* lists
  sets, 210. *See also* sets
  strings, 6. *See also* strings
`python3` command, 4, 5

Q

quadratic-time algorithm, 274, 275

R

r, mode of, 173
randint function, 257, 258
range, 88
  for loops, 59, 61, 69
  function, 59, 61
range function, 59
readline method, 174, 175
relational operators, 28–30, 78. *See also* operators
remove method, 113, 212
return
  keyword, 147
  statements, 148
  values, 8, 147
reverse keyword argument, 144
reverse=true, 112
Ribbon Painting problem, 288
  challenge, 289
  input, 289
  output, 289
  problem solving, 290
rstrip method, 175

S

School Trip problem, 118
  challenge, 118
  index of maximum, 126
  input, 119
  joining lists, 119
  list operations, 126
  list values, changing, 121
  output, 119
  problem solving, 127
  splitting strings, 119
    joining a list into string, 120
    list, into, 120
  summing a list, 126
search function, 213, 214
Secret Sentence problem, 91
  break, 96
  challenge, 91
  continue, 96
  for loops, limitations, 92
  input, 92
  output, 92
  problem solving, 95
  while loops through indices, 93
sets, 210
  add method, 212, 215
  defined, 210
  efficiency of searching, 213
  empty, 211
  intersection method, 213
  methods, 212
  mutability, 212
  of lists, 211
  remove method, 212
  update method, 213
  values in, 211
sets and dictionaries, 203
  Cities and States problem, 231
  Common Words problem, 215
  Email Addresses problem, 204
short-circuiting evaluation, 95
single quotes, 6
Ski Hills problem, 244
  challenge, 244
  input, 244
  output, 244
  problem solving, 246
    cost of range, 246
    main program, 247
slicing, 88, 89, 91, 105, 107
Slot Machines problem, 69
  Boolean operators, 78
  challenge, 70
  f-strings, 84
  for loops, limitations, 72
  input, 70
  mod operator, 82
  nesting loops in loops, 77
  output, 70
  problem solving, 79
  while loops, 73
Song Playlist problem, 86
  challenge, 86
  input, 86
  output, 87
  problem solving, 90
  string slicing, 87
sort method, 111, 112, 162, 291
sorting first, 253
  code, 253
  program efficiency, 255
split method, 120
square brackets, 88, 103, 109, 220
standard input, 44

str function, 22
strings, 6–9
  center method, 109
  count method, 8, 9, 109
  find method, 206
  isupper method, 52, 109
  join method, 120, 121
  methods, 8
  operators, 7
  representation, 6
  rstrip method, 175
  slicing, 87–90
  split method, 120
  strip method, 8
  upper method, 8, 15
  value, 6
strip method, 8
sum function, 126, 130
swaps, 48, 49
syntax error, 6

T

Telemarketers problem, 37
  Boolean operators, 38
    and, 39
    not, 40
    or, 38
  challenge, 37
  input, 37
  input and output redirection, 43, 44
  output, 38
  problem solving, 41
Terminal, 4
text editor, launching, 16
  Linux, 17
  macOS, 16
  Windows, 16
thinking, types of, xxii
Three Cups problem, 47
  challenge, 48
  input, 48
  loops, 48, 49
  nesting, 51
  output, 48
  problem solving, 54
  swaps, 54
Timsort, 281
Timus judge, xxvi, 153
TLE (time limit exceeded), 208, 265, 268
TODO markings, 157, 187
top-down design, 154–167
tuples, 224, 225
two-player card game. *See* Card Game problem
type function, 27
TypeError, 21

U

update method, 213
upper method, 8, 15
USACO (USA Computing Olympiad) judge, xxvi, 172

V

values, 6, 218
values method, 223
variables, 11, 12
  names, 13, 49
  values, 13
Village Neighborhood problem, 101
  challenge, 102
  code duplication, avoiding, 116
    huge size, 116
    list of sizes, building, 117
  input, 102
  list methods, 108, 110
    adding to, 111
    removing values from, 112
    sorting, 112
  list mutability, 106
  lists, 102, 103
  output, 102
  problem solving, 114
  size, 102

W

w, mode of, 173
while loops, 69, 73, 80, 87, 90, 91
  Boolean expression, 73, 74, 87, 96
  indefinite loops, 73
  through indices, 93
  using, 73
    Boolean operators, 78
    nesting loops in loops, 77
  variables, 73

- `while` statement, 73
- Winning Team problem, 25
  - Boolean type, 27
  - challenge, 26
  - conditional execution, 26, 27
  - `if` statements, 25, 27, 30
    - `if` by itself, 31
    - `if` with `elif`, 32
    - `if` with else, 33
  - input, 26, 35
  - output, 26, 35
  - problem solving, 35
  - relational operators, 28
- Word Count problem, 1, 2, 5
  - challenge, 5
  - input, 5
  - input-process-output model, 2
  - integer and floating-point numbers, 9
    - assignment statement, 11
    - changing variable values, 13
    - variables, 11
  - output, 5
  - problem solving, 16
    - judge, submitting to, 18
    - program, 17
    - running the program, 17
    - text editor, launching, 16, 17
  - reading input, 15
  - strings, 6
    - methods, 8
    - operators, 7
    - representation, 6
  - variable, using, 14
  - writing output, 15
- `write` method, 177–179