Symbols
+ (addition), 26
* (asterisk) operator, 147
{} (braces), 92, 105
/ (division), 26
** (double asterisk) operator, 149
== (equality operator), 72–73
** (exponent), 26
// (floor division), 260
> (greater than), 75
>= (greater than or equal to), 75
# (hash mark), for comments, 29
!= (inequality operator), 74
< (less than), 75
<= (less than or equal to), 75
% (modulo operator), 116–117, 122
* (multiplication), 26
\n (newline), 22
! (not), 74
+= operator, 115
[] (square brackets), 34
- (subtraction), 26
\t (tab), 22

A
addition (+), 26
aliases, 152
alice.py, 197–199
Alien Invasion project. See also
Pygame
aliens
  checking edges, 266
collisions, with bullets, 268–269, 291–292
collisions, with ship, 272–275
controlling fleet
direction, 266
creating an alien, 256
creating the fleet, 258–264
dropping the fleet, 267
reaching bottom of
  screen, 276
rebuilding the fleet, 270
bullets, 246–252
collisions, with aliens,
  268–269, 291–292
deleting old, 250
firing, 249
limiting number of, 251
making larger, 270
settings, 247
speeding up, 271
classes
  Alien, 256–258
  Bullet, 247–248
  Button, 280–281
  GameStats, 273
  Scoreboard, 288–289
  Settings, 231
  Ship, 233–235
ending the game, 276
files
  alien_invasion.py, 229
  bullet.py, 247
  button.py, 280
  game_stats.py, 273
  scoreboard.py, 288
  settings.py, 231
  ship.bmp, 233
initializing dynamic
  settings, 286
levels
  adding, 285–287
  modifying speed
    settings, 285
  resetting the speed, 287
planning, 228
Alien Invasion project, continued

Play button
adding, 280–285
deactivating, 284
drawing, 281
hiding the mouse
cursor, 284
resetting the game, 283
starting the game, 283
scoring, 288–300
all hits, 292
high score, 294
increasing point values, 292
level, 296–298
number of ships, 298–300
resetting, 291
rounding and
formatting, 293–294
score attribute, 288
updating, 291
settings, storing, 231
ship
adjusting speed, 241–243
continuous movement, 239–241
finding an image, 232
limiting range, 243

amusement_park.py, 80–83

and keyword, 75

API (application programming interface), 359
calls, 359
for GitHub, 371
for Hacker News, 372–375
processing responses, 361–365
rate limits, 365
requesting data, 360
visualizing results, 366–371

apostrophe.py, 24
append() method, 37

application programming interface. See API (application
programming interface)
arguments, 131–137. See also functions: arguments
arithmetic, 26
as keyword, 152
assert methods, 212, 216

asterisk (*) operator, 147
attributes, 159. See also classes: attributes

B

banned_users.py, 77
bicycles.py, 34–36
.bmp (bitmap) image files, 232

body
of a function, 130
of an HTML file, 440

Boolean values, 77, 456
Bootstrap, 438–447
braces ({}), 92, 105
built-in functions, 471

C

CamelCase, 181
car.py, 162–179
cars.py, 43–45, 72
cities.py, 121
classes
attributes, 159
accessing, 160
default values, 163
modifying, 164–166
creating, 158–162
importing, 174–179
all classes from a
module, 177
multiple classes, 175–177
single class, 174–175
inheritance, 167–173
attributes and methods, 169
child classes, 167
__init__() method, 167–169
instances as attributes,
170–172
overriding methods, 170
parent classes, 167
subclasses, 168
super() function, 168
superclasses, 168
instances, 157
methods, 159
calling, 160
__init__() method, 159
modeling real-world objects, 173
multiple instances, 161
naming conventions, 159
objects, 157
styling guidelines, 181
comma-separated value files. See
CSV (comma-separated value) files
comment.py, 29
comments, 29–30
conditional tests, 72–77. See also
if statements
confirmed_users.py, 124
constants, 28
counting.py, 118, 122
CSV (comma-separated value) files, 334–346
error-checking, 343–345
parsing headers, 334–335
reading data, 336
data analysis, 305
databases. See Django: databases
data visualization, 305. See also
Matplotlib
datetime module, 337–339
depth_valley_highs_lows.py, 343–345
decorators, 429
default values, 134
class attributes, 163
function parameters, 134
def keyword, 130
del statement, 39
dice.py, 324
dice_visual.py, 325–327
dimensions.py, 66–67
Discord, 48
div (HTML), 441
division (/), 26
division_calculator.py, 194–197
Django, 379. See also Heroku;
Learning Log project
admin site, 387–392
registering models, 388, 391
associating data with users, 435
Bootstrap, 438–447
card, 446
collapsible navigation, 440
container element, 443
django-bootstrap4 app, 438
HTML headers, 439–440
jumbotron, 443
navigation bar, 440–442
styling forms, 444–445
commands
createsuperuser, 388
flush, 433
makemigrations, 387, 391, 432
migrate, 383
shell, 392
startapp, 385, 421
startproject, 382
creating a project, 381
databases
cascading delete, 390
creating, 382
foreign keys, 390
many-to-one
relationships, 390
migrating, 383, 391
queries, 404, 433
querysets, 392–393, 401
required (non-nullable)
fields, 432
Django, continued
databases, continued
  resetting, 433
  SQLite, 383
decorators, 429
deployment. See Heroku
development server, 383, 389
documentation
  Django, 379
  models, 386
  queries, 394
  templates, 406
forms, 410–420
  action argument, 413
cross-site request
displaying, 413
GET and POST requests, 412
ModelForm, 410, 414
pre-filling with data, 419
processing, 412, 416
validation, 410
widgets, 414
get_object_or_404() method, 460
hashes (for passwords), 388
HTML
  div element, 441
  main element, 442
  margin, 443
  padding, 443
  span element, 442
HTTP 404 error, 434
INSTALLED_APPS, 386, 421, 438
installing, 381
localhost, 383
logging out, 424
login page, 422
@login_required, 429
login template, 422
mapping URLs, 395–396
migrating the database,
  383, 391
models, 385–393, 431
privileges, 387
projects (vs. apps), 384
redirect() function, 411
registration page, 426–428
release cycle, 381
restricting access to data,
  433–435
restricting access to pages,
  428–435
settings.py
  INSTALLED_APPS, 386, 421, 438
  LOGIN_URL, 429
  SECRET_KEY, 461
shell, 392, 431
starting an app, 385
starting a new project, 382
static files, 448
styling. See Django: Bootstrap
superuser, 387
templates, 397
anchor tags, 399
block tags, 399
context dictionary, 401
filters, 405
for loop, 402
indentation in, 399
inheritance, 398
linebreaks filter, 405
template tags, 399
third-party apps, 438
URLs
  capturing values, 404
  namespaces, 399
  URL patterns, 395–396
  url template tag, 399
user ID values, 431
users
  default login view, 422
displaying message to
    logged-in user, 424
  logging in a user, 427
    UserCreationForm, 427
versions, 381
views, 396
retrieving objects, 401, 404
docstrings, 130
dog.py, 158–162
dot notation, 151, 160
double asterisk (**) operator, 149
E

earthquakes. See mapping earthquakes
electric_car.py, 168–173
electric_car.py module, 178
enumerate() function, 335
environment variables, 456
ePOCH time, 366
eq_explore_data.py, 348–351
equality operator (==), 72–73
eq_world_map.py, 351–357
even_numbers.py, 58
even_or_odd.py, 117
exceptions, 183, 194–202
  deciding which errors to report, 201
  else blocks, 196
  failing silently, 200
  FileNotFoundError, 197
  handling, 194
  try-except blocks, 194
  using to prevent crashes, 195
  ZeroDivisionError, 194
exponents (**), 26

F

favorite_languages.py, 97–98
FileNotFoundError, 197
file_reader.py, 184–188
files
  closing, 185
  file paths, 185
    absolute, 186
    relative, 186
  opening, 184
    append mode, 192
    read mode, 192
    write mode, 192
  reading from, 184–190
    entire files, 184–185
    line by line, 187
    making a list of lines, 188
    working with contents, 188
    working with large files, 189
writing to
  by appending, 193
  empty files, 191
  multiple lines, 192
first_numbers.py, 57
flags, 120
floats, 26
foods.py, 63–65
for loops, 49. See also dictionaries; lists
formatted_name.py, 138–140
full_name.py, 21–22
functions, 129
  alias (as), 152
  arguments, 131–137
    arbitrary keyword, 148
    arbitrary number of, 147
    avoiding errors, 136
    keyword, 133
    lists as, 143–146
    mixing positional and arbitrary, 148
    optional, 138
    order of, 133
    positional, 132–133
  built-in, 471
  calling, 130–137
    equivalent calls, 135–136
    multiple times, 132
  defining, 130
  dictionaries, returning, 140
  lists in
    modifying, 143–145
    protecting, 145
  modules, 150–155
    alias (as), 153
    importing all functions (*), 153
    importing entire modules, 150
    importing specific functions, 152
  parameters, default values for, 134
  passing information to. See functions: arguments
  return values, 137–142
  styling, 154
games. See Alien Invasion project; Pygame
Geany, 476
GET requests, 412
getting help
Discord, 484
IRC (Internet Relay Chat), 482–483
official Python documentation, 481
online resources, 480
r/learnpython, 482
rubber duck debugging, 480
Slack, 483
Stack Overflow, 481
three main questions, 479
Git, 360, 450
branches, 452, 488
commits, 360, 450
checking out, 491–493
making, 452, 457, 460, 488, 490, 493
configuring, 451, 486
detached HEAD, 492
files
adding, 452, 460, 488, 493
ignoring, 451, 487
HEAD, 492
installing, 450, 486
log, checking, 489
repositories, 360
deleting, 493
initializing, 452, 487, 493
reverting changes, 490
status, checking, 452, 457, 487–493
GitHub, 360
greater than (>), 75
greater than or equal to (>=), 75
greeter.py, 114, 130–131
greet_users.py, 143
gunicorn package, 448

Hacker News, 372
hash mark (#), for comments, 29
head, of an HTML file, 440
HEAD (Git), 492
Hello World, 9
hello_git.py, 486–491
hello_world.py, 10, 15–19
Heroku, 437. See also Django; Git;
Learning Log project
Bash shell, 454
CLI, installing, 448
commands
config, 458
destroy, 462
login, 453
open, 453
ps, 453
rename, 455
run, 454
set, 458
databases, setting up, 454
django-heroku package, 448
documentation, 453
environment variables, setting, 456–458
error pages, custom, 458–460
free plan, limitations of, 448, 456
making an account, 448
Procfile, 450
projects
deleting, 461
pushing to Heroku, 452–453, 457
viewing live, 453
Python runtime, specifying, 449
requirements.txt, 448–449
securing, 456
settings.py, modifying for, 450, 456, 459
superuser, creating, 454–455
URLs, user-friendly, 455
hidden files, 451
hn_article.py, 372
hn_submissions.py, 373
Homebrew, 469

IDE (integrated development environment), 473
IDLE, 475
if statements
   and keyword, 75
   Boolean expressions, 77
   checking for
equality (==), 72
   equality, ignoring case, 73
   empty lists, 87
   inequality (!=), 74
   items in a list, 76
   items not in a list, 77
   conditional tests, 72–77
   elif statement, 80–84
   else statement, 79
   lists and, 85–88
   numerical comparisons, 74–76
   or keyword, 76
   simple, 78
   styling guidelines, 90
   testing multiple conditions, 83–84
immutable, 65
import *, 153
import this, 30
indentation errors, 53–56
index errors, 47
inequality operator (!=), 74
infinite loops, 122
inheritance, 167. See also classes: inheritance
input() function, 114–117
   numerical input, 115–116
   prompts, 114
insert() method, 38
IRC (Internet Relay Chat), 482–483
itemgetter() function, 374
items() method, 100

J
JSON files
   examining data, 347
geoJSON file format, 349
json.dump() function, 348
json.load() function, 204
jumbotron, 443

K
keys() method, 101
key-value pairs, 92–99. See also
dictionaries
keyword arguments, 132, 133. See also functions
keywords, 471

L
language_survey.py, 217
Learning Log project, 379. See also
   Django; Heroku
deployment. See Git, Heroku files
404.html, 458
500.html, 459
admin.py, 388
base.html, 399, 403, 423, 425, 428, 439–443
edit_entry.html, 419
forms.py, 410, 414
.gitignore, 451
index.html, 397, 443
learning_logs/urls.py, 395, 401, 404, 411, 415, 418
learning_log/urls.py, 395, 422
logged_out.html, 425
login.html, 422–423, 444
models.py, 385, 390, 431
new_entry.html, 416
new_topic.html, 412
Procfile, 450
register.html, 427
requirements.txt, 448
runtime.txt, 449
settings.py, 386, 421, 429, 450, 456, 459
Learning Log project, continued
files, continued

topic.html, 404–405, 417, 420
topics.html, 402, 405, 413, 445
users/urls.py, 422, 426
users/views.py, 426
views.py, 396, 401, 404,
411, 415, 418, 429, 430,
433, 460

pages
edit entry, 418–420
home, 394–398
logout, 424
new entry, 414–417
new topic, 410–413
register, 426
topic, 403
topics, 400–403
users app, 421–428
virtual environment, 380–381
writing a specification
(spec), 380

len() function, 45
less than (<), 75
less than or equal to (≤), 75

Linux
Hello World, running, 10
Python
checking installed version, 8
installing, 470
troubleshooting
installation, 11
Sublime Text, installing, 9
terminal
running programs from, 12
starting Python session, 9

lists, 33
as arguments, 143–146
copying, 63–65
elements
accessing, 34
accessing last, 35
adding with append(), 37
adding with insert(), 38
identifying unique, 104
modifying, 36
removing with del, 39
removing with pop(), 39–41
removing with remove(), 41
empty, 38
enumerate() function, 335
for loops, 49–56
nested, 109, 263
if statements and, 85–88
indentation errors, 53–56
indexes, 35
time, 46–47
negative, 35
len() function, 45
list comprehensions, 59
naming, 34
numerical lists, 57–60
range() function, 58–59
removing all occurrences of a
value, 125
slices, 61–63
sorting, 43–46
reverse() method, 45
sorted() function, 44
sort() method, 43
logical errors, 54

M

macOS
Hello World, running, 10
Python
checking installed version, 7
installing, with Homebrew,
469–470
installing, official
version, 7–8
troubleshooting
installation, 11
Sublime Text, installing, 8
terminal
running programs from, 12
starting a Python session, 8
magicians.py, 50–53
magic_number.py, 75
making_pizzas.py, 151–153
mapping earthquakes, 347–357
building a world map, 351
colorscales, 354
data
downloading, 347, 358
examining JSON data, 347
extracting locations, 351
extracting magnitudes, 350
geoJSON file format, 349
hover text, 356
latitude-longitude ordering, 349
magnitudes, representing, 353
Scattergeo chart type, 352
Matplotlib, 306–323, 336–347
axes
axis() method, 313
ax variable, 307
removing, 321
fig variable, 307
formatting plots
alpha argument, 342
built-in styles, 310
colormaps, 314
custom colors, 314
labels, 307–308
line thickness, 307
shading, 342
size, 322
gallery, 306
installing, 306
line graphs, 306
plot() method, 307, 309
pyplot module, 307
saving plots, 315
scatter plots, 310–313
subplots() function, 307
methods, 20, 236–237. See also classes
modules, 150. See also classes:
modules; functions:
modules
modulo operator (%), 116–117, 122
motorcycles.py, 37–42
mountain_poll.py, 126
mpl_squares.py, 306–310
multiplication (*), 26
my_car.py, 175
my_cars.py, 177–179
my_electric_car.py, 176
N
name errors, 17
name_function.py, 210–215
name.py, 20
names.py, 210
nesting, 106–112
depth, 110
dictionaries in dictionaries,
110–111
dictionaries in lists, 106–108
lists in dictionaries, 108–110
newline (\n), 22
None, 99
not (!), 74
number_reader.py, 203
numbers, 25–28
arithmetic, 26
comparisons, 74–76
exponents, 26
floats, 26
floor division (\//), 260
integers, 26
mixing integers and floats, 27
order of operations, 26
round() function, 294
underscores in, 28
number_writer.py, 203
O
object-oriented programming, 157.
See also classes
open() function, 184
or keyword, 76
P
parameters, 131
parent classes, 167. See also classes:
inheritance
parrot.py, 114, 118–121
pass statement, 200
PEP 8, 68–70, 90, 154
person.py, 140–142
pets.py, 125, 132–136
pip, 228
  installing Django, 381
  installing Matplotlib, 306
  installing Plotly, 324
  installing Pygame, 228
  installing Requests, 361

pi_string.py, 188–190
pizza.py, 147–148
players.py, 61–63

Plotly
  Bar() class, 327
  data, long format, 353
  formatting plots
    bars, 368
    colorscales, 354
    hover text, 356, 369–370
    marker color, 368
    marker size, 353
    layout() class, 327
    layout, long format, 367
    x-axis, 368
    y-axis, 368
  gallery, 324
  histogram, 326
  installing, 324
  offline.plot() function, 327
  Python figure reference, 371
  Scattergeo chart type, 352
  user guide, 371

positional arguments. See functions: arguments

POST requests, 412

printing_models.py, 143–146

Project Gutenberg, 198

.py file extension, 16

Pygame. See also Alien Invasion project
  background colors, 230–231
  collisions, 268–269, 272–275, 291–292
  colors, 231
  creating empty windows, 229
cursor, hiding, 285
displaying text, 280
drawing all elements in, 259
ending games, 276
event loops, 230

fullscreen mode, 244
groups, 248
  drawing all elements in, 259
  emptying, 271
  looping through, 250
  removing elements from, 251
  storing elements in, 248
  updating all elements in, 249
images, drawing to screen, 235
images, loading, 234
installing, 228
print() calls in, 251
quitting, 244
rect objects
  creating from scratch, 247
  responding to input, 230
  keypresses, 238
  mouse clicks, 283
  screen coordinates, 234
  surfaces, 230
testing games, 270

pyplot module, 307

Python
  >> prompt, 4
  built-in functions, 471
documentation, 481
installing
  on Linux, 470
  on macOS, official, 7–8
  on macOS, using Homebrew, 469–470
  on Windows, 5–6, 467–469
interpreter, 16
keywords, 471
PEP 8, 68–70, 90, 154
standard library, 180–181
terminal sessions, 4
versions, 4
Zen of, 30–31

Python Enhancement Proposal (PEP), 68

python_repos.py, 361–365
python_repos_visual.py, 366–371
Q
quit values, 118–120

R
random_walk.py, 316–317
random walks, 315–323
  choice() function, 317
  coloring points, 319
  fill_walk() method, 316
  multiple walks, generating, 318
  plotting, 317
  RandomWalk class, 316
  starting and ending points, 320
range() function, 57–59
readlines() method, 188
read() method, 185
refactoring, 206–208, 236
remember_me.py, 204–208
Requests package, 361
return values, 137
r/learnpython, 482
rollercoaster.py, 116
rolling dice, 323–330
  analyzing results, 325
  Die class, 324
  different sizes, 329
  randint() function, 324
  two dice, 328
rubber duck debugging, 480
rw_visual.py, 317–323

S
scatter_squares.py, 311–315
sets, 104
sitka_highs_lows.py, 340–342
sitka_highs.py, 334–340
Slack, 483
sleep() function, 274
slice, 61
split() method, 198
square brackets ([[]]), 34
squares.py, 58, 60
Stack Overflow, 481
storing data, 202–205. See also JSON
strings, 19–25
  changing case, 20
  format() method, 22
  f-strings, 21
  newlines in, 22
  single and double quotes, 19, 24
  tabs in, 22
  using variables in, 21
  whitespace in, 22–24
strptime() method, 338
style guidelines, 68–70
  blank lines, 69
  CamelCase, 181
  classes, 181
  functions, 154
  if statements, 90
  indentation, 69
  line length, 69
  PEP 8, 68
Sublime Text, 4–10, 474–475
  commenting out code, 475
  configuring, 9
  customizing, 474
  indenting and unindenting code blocks, 474
  installing, 7–9
  line length indicator, 474
  running Python programs, 9–10
  saving your configuration, 475
  tabs and spaces, 474
subtraction (-), 26
superclasses, 168. See also classes: inheritance
survey.py, 217
syntax errors, 24
syntax highlighting, 16

T
tab (\t), 22
testing code, 209–222
  adding tests, 214
  assert methods, 216
  coverage, 211
  failing tests, 212–214
  passing tests, 211–212
  setUp() method, 220
  test case, 211
  testing classes, 216–221
  testing functions, 210–215
  unittest module, 209
  unit tests, 211
text_name_function.py, 211–215
test_survey.py, 218–221
text editors and IDEs
  Atom, 476
  Emacs and Vim, 476
  Geany, 476
  IDLE, 475
  Jupyter Notebooks, 477
  PyCharm, 476
  Sublime Text, 4–10, 474–475
  Visual Studio Code, 476
toppings.py, 74, 83–88
traceback, 18
try-except blocks, 194–202. See also exceptions
tuples, 65–67
type errors, 66

U
unittest module, 209
unit tests, 211
Unix time, 366
user_profile.py, 148

V
values() method, 104
variables, 16–19, 28
  constants, 28
  as labels, 18
  multiple assignment, 28
  naming conventions, 17
  values, 16
version control system, 485.
  See also Git
virtual environment (venv), 380
voting.py, 79–80

W
weather data, 334–347
web framework, 379
while loops, 118–127
  active flag, 120–121
  break statement, 121
  infinite, 122
  moving items between lists, 124
  quit values, 118–120
whitespace, 22–24
Windows
  Hello World, running, 10
  Python
    checking installed version, 5
    installing, 5–6, 467–469
    troubleshooting installation,
      11, 467–469
  Sublime Text, installing, 7
  terminal
    running programs from, 12
    starting a Python session, 6
with statement, 185
word_count.py, 199–201
write_message.py, 191–193
write() method, 192

Z
Zen of Python, 30–31
ZeroDivisionError, 194