

# INDEX

Note: Page numbers followed by *f*, *n*, or *t* indicate figures, notes, and tables, respectively.

## SYMBOLS

`+=` (addition and assignment operator), 100, 187  
`+` (addition operator), 35, 35t  
`\` (backslash), 240  
`/` (division operator), 35, 35t, 39  
`=` (equal sign), 32  
`==` (equal to operator), 62, 79–80, 82–83, 82t  
`>` (greater than operator), 62, 82–83, 82t  
`>=` (greater than or equal to operator), 82–84, 82t  
`()` (grouping operator), 35, 35t  
`#` (hash mark), 6  
`//` (integer division operator), 49–50, 114, 156  
`<` (less than operator), 62, 82–84, 82t  
`<=` (less than or equal to operator), 82–83, 82t  
`%` (modulo [mod] operator), 21–22, 49–50, 88–89, 156  
`*` (multiplication operator), 35, 35t  
`!=` (not equal to operator), 62, 82t, 83  
`**` (power [exponent] operator), 35t  
`"` (quotation marks), 32  
`'` (single quotation marks), 80

`[]` (square brackets), 46  
`-` (subtraction operator), 35, 35t

## A

addition and assignment operator (`+=`), 100, 187  
addition operator (`+`), 35, 35t  
algorithms, defined, 20, 41, 296  
American Standard Code for Information Interchange (ASCII) values, 97, 98t  
and (logical operator), 93–94, 93t  
animation, 175–206. *See also* drawing  
bouncing  
  changing direction, 194–196  
  off four walls, 197–201  
  off one wall, 190–197  
  speed, 194–199  
boundaries, 191–193, 198  
collision detection, 191, 195, 198, 200, 221, 224–225, 237–240, 238f, 247–248  
defined, 296  
frames, 185–186  
game loop, 182–184  
movement, 186–190  
Pygame, 176–181  
`append()` function, 66  
appending, 65, 97, 296  
applications (apps; programs), defined, 2, 296

arguments, 43, 110, 146, 296  
arrays, 119–122, 122t, 126–128  
  defined, 119, 296  
*ArrowDraw.py* program, 161–163, 163f, 173  
ASCII (American Standard Code for Information Interchange) values, 97, 98t  
assignment, of values to variables, 32  
  defined, 296  
*AtlantaPizza.py* program, 39–42, 42f  
attributes, defined, 216

## B

backslash (`\`), 240  
`bgcolor()` function, 23, 159  
binary search, 108  
BLACK variable, 188  
`blit()` function, 183, 188, 243, 256  
BMP format, 182  
Boolean (conditional) expressions, 62, 79, 81–85  
  comparison operators, 81–84  
  defined, 296  
Boolean numbers, 34  
bouncing (animation)  
  changing direction, 194–196  
  off four walls, 197–201  
  off one wall, 190–197  
  speed, 194–199  
boundaries, for animations, 191–193, 198

## C

Caesar cipher, 95, 95f  
callback functions, 158  
calling functions,  
    defined, 144  
canvas size, determining,  
    113–114  
Cartesian coordinates,  
    111–112, 111f,  
    133, 133f  
case sensitivity, 33  
characters, 97–99  
choice() function, 110, 115,  
    116, 120–121  
chr() function, 100  
ciphers, 95–100  
circle() function, 17–19,  
    54–55, 57, 148,  
    179, 183  
*CircleSpiral1.py* program,  
    17–19, 18f  
*CircleSpiralInput.py*  
    program, 52  
classes, 216–219. *See also*  
    *names of specific*  
    *classes*  
    constructed, 218  
    container, 217  
    defined, 189, 296  
    extending, 218  
*ClickAndSmile.py* program,  
    166–167, 167f, 173  
*ClickDots.py* program,  
    208–211, 209f  
*ClickKaleidoscope.py*  
    program, 170–171,  
    171f, 173  
*ClickSpiral.py* program,  
    163–165, 164f  
close window button (event),  
    182–184  
Clock class, 188–190  
coding, defined, 1  
collide\_circle()  
    function, 224  
collidepoint() function, 224

collide\_rect() function, 224  
collision detection, 191,  
    195, 198, 200, 221,  
    224–225, 237–240,  
    238f, 247–248  
    defined, 296  
*ColorCircleSpiral.py*  
    program, 23–24, 52  
*ColorMeSpiralled.py*  
    program, 52  
*ColorPaint.py* program, 229  
colors, 19  
    changing background, 23  
    using multiple, 20–22  
colors argument, 110  
colorspiral module  
    building, 290–291  
    reusing, 292–293  
    using, 291–292  
*ColorSpiralInput.py*  
    program, 47–48,  
    48f, 52  
*ColorSpiral.py* program, 25,  
    26f, 27–28  
*ColorSquareSpiral.py*  
    program, 21–22, 21f  
colors variable, 20–21, 46  
comments, 13  
    defined, 6  
    docstrings, 291  
    usefulness of, 40–41  
comparison operators,  
    62–63, 81–84,  
    82t, 83f  
complex conditions, 92–94  
complex numbers, 34  
compound if statements, 93  
concatenation, 97, 296  
conditional expressions,  
    defined, 296.  
    *See also* Boolean  
    expressions;  
    conditions  
conditions, 77–103  
    Boolean expressions, 62,  
    79, 81–85  
ciphers, 95–100

complex, 92–94  
elif statements, 91–92,  
    118, 253  
else statements, 85–91  
if statements. *See* if  
    statements  
    while statements, 62–64  
constants, 179, 188, 296  
constructors, 218  
container classes, 217  
convert\_in2cm() function,  
    153–156  
convert\_lb2kg() function,  
    154–156  
count\_popped variable,  
    254–256  
count\_smileys variable,  
    254–256  
cspiral() function, 291–293

## D

declaring (defining)  
    functions, 143–144,  
    150–151  
    defined, 296  
def keyword, 143  
Descartes, René, 111  
diameter, 19  
*DiscoDot.py* program, 203  
division operator (/), 35,  
    35t, 39  
docstrings, 291  
downloading Python, 4–5  
*DragDots.py* program,  
    211–214, 211f, 228  
drawing, 11–29. *See also*  
    animation  
    circles, 17–19  
    colors, 19  
        changing  
        background, 23  
        using multiple, 20–22  
    dots, 177–180  
    multi-sided spirals, 25–26  
    square spirals  
        adjusted, 16–17  
        basic, 12–15

`draw_kaleido()` function, 168–169, 171  
`draw_smiley()` function, 146, 150–151, 166  
`draw_spiral()` function, 169–171  
`draw_string` variable, 242  
`driving_age` variable, 85f

## E

element, defined, 297  
elif statements, 91–92, 118, 253  
else statements, 85–91  
*EncoderDecoder.py* program, 99–100, 102–103  
`end_fill()` function, 148  
end keyword, 43  
equal sign (=), 32  
equal to operator (==), 62, 79–80, 82–83, 82t  
`eval()` function, 28, 40, 48  
evaluation, 28, 40  
event, defined, 297  
event handlers (event listeners), 157–158, 160–165, 183–184  
close window button, 182–184  
defined, 157  
keyboard events, 160–163, 246–247  
mouse clicks, 158–160, 163–171, 209–210  
mouse presses and releases, 213  
parameters, 163–171  
in Pygame, 181  
exponent (power) operator (\*\*), 35t  
expressions  
  Boolean, 62, 79, 81–85  
  defined, 36, 297  
  in shell, 36, 36f  
extending classes, 218

## F

False value, 83–84  
`fillcolor()` function, 148  
`fill()` function, 148, 188  
*FiveDice.py* program, 129–131, 130f  
flags, 123–124, 212, 225  
floating-point numbers, defined, 34–35  
focus, 162  
fonts (typefaces), 242–243  
for loops, 14, 55–59, 65, 80, 135, 142, 144, 179, 214  
  defined, 297  
`forward()` function, 13–15, 17, 142–143  
frames, 185–186, 297  
frames per second (fps), 186, 297  
functions, 141–173. *See also names of specific functions*  
  callback, 158  
  calling, 144  
  defined, 17, 297  
  defining, 143–144, 150–151  
  interaction, 157–171  
  parameters, 146  
  returning values from, 153–154  
  using return values in programs, 154–157  
  utility of, 142

## G

game keycodes, 247  
game loops, 61–62, 108, 130, 135, 181  
  animation, 182–184  
  handling mouse clicks, 209–210  
  handling mouse presses and releases, 213  
  ongoing play, 123–124

game programming, 231–262. *See also names of specific games*  
  adding difficulty, 247–249  
  adding points, 240–241  
  board and pieces, 234–241  
  displaying score, 241–245  
  elements of design, 232  
  game over, 246  
  hitting ball with paddle, 237–240  
  playing again, 246–247  
  sound, 252–254  
  subtracting lives, 236–237  
  tracking and displaying progress, 254–257  
`get()` function, 181, 210  
`get_height()` function, 193, 199, 221  
`get_pos()` function, 213, 235  
`get_pressed()` function, 253  
`get_rect()` function, 243  
`get_rel()` function, 229  
`get_width()` function, 193, 195, 221  
`goto()` function, 150  
greater than operator (>), 62, 82–83, 82t  
greater than or equal to operator (>=), 82–84, 82t  
GREEN variable, 178–179  
Group class, 217–218  
grouping operator (()), 35, 35t  
*GuessingGame.py* program, 107–109, 108f  
guess variable, 108

## H

hash mark (#), 6  
`heading()` function, 69–70, 139

- height\_cm variable, 156
  - height\_in variable, 156
  - "Hello, world!" program, 5
  - hideturtle() function, 152
  - HighCard.py* program, 125, 139
  - Hi-Lo guessing game, 106–109
- I**
- IDLE editor, 5, 269–270, 275–276, 277–278, 282–283, 286
  - if-elif-else statements, 91–92, 128
  - if-else statements, 85–91
  - IfSpiral.py* program, 79–81, 81f
  - if statements, 78–81, 78f, 239–241, 247–248
    - defined, 78
    - games, 108, 124
    - syntax, 79
  - importing code, defined, 13, 297
  - indentation, 56
  - indexes
    - finding items in lists, 121–122
    - testing which value is higher, 122–123
  - index() function, 122
  - init() function, 178, 182, 208, 253
  - \_\_init\_\_() function, 218–220, 223
  - initialization, 178, 218, 297
  - in keyword, 56
  - inner and outer loops, 68
  - input, defined, 297
  - input() function, 33, 92, 94, 142–143
  - installing Pygame
    - for Linux, 287–288
    - for Mac, 284–286
    - for Windows, 280–283
  - installing Python, 5
    - for Linux, 276–278
    - for Mac, 271–274
    - for Windows, 264–268
  - integer division operator (//), 49–50, 114, 156
  - integers, defined, 34
  - interaction, 207–229. *See also* event handlers (event listeners)
    - classes and objects, 216–219
    - clicking, 208–211
    - collision detection, 224–225
    - dragging, 211–214
    - sprites, 215–216
      - removing, 224–225
      - scaling, 221
    - setting up, 218–220
    - updating, 220–221
  - int() function, 60
  - islower() function, 97
  - isupper() function, 96, 100
  - iteration, 55
  - iterative versioning, defined, 252, 297
- K**
- kaleidoscope mirror effect, 132–136, 139
  - Kaleidoscope.py* program, 134–135, 136f, 138–139, 168
  - keep\_going variable, 123–125, 178, 182–183, 209
  - keyboard events, 160–163, 246–247
  - keycodes, Pygame, 247
  - KEYDOWN event, 246–247
  - keyword arguments, 43, 297
- L**
- left() function, 13–14, 16, 18, 28, 55, 57–58, 161
  - len() function, 66–67, 253, 255
  - less than operator (<), 62, 82–84, 82t
  - less than or equal to operator (<=), 82–83, 82t
  - libraries, defined, 13
  - listen() function, 162
  - listeners. *See* event handlers (event listeners)
  - list() function, 55–56
  - lists, 46–48, 120
    - defined, 46, 297
    - finding items in, 121–122, 122t
  - load() function, 182–183
  - logical operators, 93, 93t
  - Logo programming language, 13n
  - loops, 53–75
    - defined, 13–14, 297
    - for, 14, 55–59, 65, 80, 135, 142, 144, 179, 214, 297
    - game. *See* game loops
    - nested, 68–72, 298
    - repeating variables through, 42
    - user input, 59–61
    - using to cycle through colors, 21–22
    - while, 62–64, 65–66, 78, 108, 123–125, 130, 179, 214, 235, 299
  - lower() function, 92–94, 96
- M**
- MadLib.py* program, 8–9
  - MadLib2.py* program, 9
  - math, in the Python shell, 36, 36f. *See also* numbers
  - MathHomework.py* program, 48–50, 50f
  - methods, defined, 216
  - mobile apps, 257

modules. *See also names of specific modules*

- building, 290–291
- defined, 298
- reusing, 292–293
- using, 291–292

modulo (mod) operator (%),  
21–22, 49–50,  
88–89, 156

MOUSEBUTTONDOWN event,  
210–211, 224–225

MOUSEBUTTONUP event, 212

mouse clicks (events),  
158–160, 163–171,  
209–210

mouse presses and releases  
(events), 213

movement (animation),  
186–190

multiplication operator (\*),  
35, 35t

multi-sided spirals, 25–26  
*MultiSpiral.py* program,  
291–292, 292f

my\_face variable, 123

my\_name variable, 32–33

## N

name variable, 42

nested loops, 68–72, 298  
*NiceHexSpiral.py* program,  
2f–3f

not (logical operator), 93, 93t

not equal to operator (!=),  
62, 82t, 83

number\_of\_circles  
variable, 60

numbers, 34–42

- Booleans, 34
- complex, 34
- doing math in shell,  
36, 36f
- floating-point, 34–35
- integers, 34
- operators, 35
- syntax errors, 37–38, 37f

numinput() function, 46–47,  
59–60, 69

## O

object-oriented  
programming, 216

objects, defined, 298

*OldEnoughOrElse.py*  
program, 86

*OldEnough.py* program,  
84–85, 85f

onkeypress() function,  
161–162

onclick() function,  
158, 160, 163–166,  
168–169, 173

operators

- comparison, 62–63,  
81–84, 82t, 83f
- defined, 35, 298
- logical, 93, 93t
- math, 35t
- programming with, 39–42

or (logical operator), 93, 93t

ord() function, 99–100

origin, 111, 180

## P

parameters, 146–153

- defined, 298
- for event handlers,  
163–171

pencolor() function, 20–21,  
46, 115, 148, 159

pendown() function, 46, 112

penup() function, 66, 69, 112

picx variable, 186–187,  
192–193, 193f,  
194–195, 197–198

picy variable, 186–187,  
192–193, 193f,  
194–195, 197–198

*PingPongCalculator.py*  
program,  
154–157, 173

ping\_pong\_heavy  
variable, 156

ping\_pong\_tall variable, 156

pixels, defined, 15, 298

*PolygonOrRosette.py*  
program, 86–87, 87f

Pong (game), 232–233, 233f

porting code, defined, 145

position() function, 69–70

pos parameter, 219–220, 225

power (exponent) operator  
(\*\*), 35t

print() function, 43, 63, 156

programming languages,  
defined, 3–4

programs (applications;  
apps), defined,  
2, 298

prompt  
command, 5  
input, 33

prototypes, defined, 252

pseudorandom numbers,  
107, 298

Pygame, 176, 176f, 180f

- classes and objects, 216
- Clock class, 189
- collision detection, 224
- event handlers, 181
- exiting program, 184
- game loop, 181–184
- get\_height() function,  
193, 199, 221
- get\_width() function, 193,  
195, 221
- Group class, 217
- initializing, 178
- installing, 177
  - for Linux, 287–288
  - for Mac, 284–286
  - for Windows, 280–283
- keycodes, 247
- scale() function, 221
- setup, 181
- sound, 252–253
- Sprite class, 216–218, 224

Pygame, *continued*  
 surfaces, 178  
 turtle graphics vs.,  
 180–181, 214  
 update() function, 220  
 pygame.draw module, 179, 183  
 pygame module, 177–179, 182  
 Python  
 defined, 4  
 documentation for, 294  
 downloading, 4–5  
 installing, 5  
 for Linux, 276–278  
 for Mac, 271–274  
 for Windows, 264–268  
 setup  
 for Linux, 278  
 for Mac, 275–276  
 for Windows, 269–270  
 website, 4f, 5

## Q

QUIT event, 183, 210  
 quit() function, 179,  
 184, 214  
 quotation marks ("), 32

## R

radius, 17–18  
*RainingDots.py* program,  
 205, 206f  
 randint() function, 106–107,  
 109, 115  
*RandomDots.py* program,  
 203–205, 204f  
 random module, 106  
 choice() function, 110,  
 115, 116, 120–121  
 importing, 106  
 randint() function,  
 106–107, 109, 115  
 randrange() function,  
 114–115, 152  
 randomness, 105–139  
 Hi-Lo guessing game,  
 106–109

kaleidoscope mirror  
 effect, 132–136  
 random spirals, 109–116  
 choosing random  
 colors, 110  
 coordinates, 111–112  
 determining canvas  
 size, 113–114  
 Rock-Paper-Scissors  
 game, 116–118  
 War-style card game,  
 119–125  
 building deck of cards,  
 119–120  
 continuing play,  
 123–125  
 counting cards,  
 121–123  
 dealing cards, 120–121  
 Yahtzee-style game,  
 126–132  
 probabilities, 131–132  
 setting up, 126–127  
 sorting dice, 127–128  
 testing dice, 128–129  
*RandomPaint.py*  
 program, 228  
*RandomSmileys.py* function,  
 146–152, 146f,  
 153f, 166  
 random\_spiral() function,  
 164–165  
 calling, 144–145  
 defining, 143–144  
*RandomSpiralsFunction.py*  
 program,  
 143–145, 164  
*RandomSpirals.py* program,  
 109–116, 109f, 132,  
 142–143  
 randrange() function,  
 114–115, 152  
 range, defined, 298  
 range() function, 17, 23,  
 55–56, 58, 59–60  
 remove() function,  
 224–225, 253

render() function, 242  
 returning (to new line), 44  
 return statement, 154  
 return values, 153–157  
 reusing code, 13, 142–143,  
 145, 189, 216–217,  
 292–293  
 RGB color triplets, 178,  
 188, 203, 298  
 right() function, 161  
 Rock-Paper-Scissors game,  
 116–118  
*RockPaperScissors.py*  
 program,  
 116–118, 118f  
*Rosette4.py* program, 56–57  
*Rosette6.py* program, 58–59,  
 59f, 74  
*RosetteGoneWild.py*  
 program, 60–61,  
 61f, 74  
*Rosette.py* program, 54, 54f  
*RosettesAndPolygons.py*  
 program, 88–89, 90f,  
 91, 102  
 rotations, 95  
 round() function, 156, 256  
*RubberBandBall.py*  
 program, 28, 29f  
 running programs, 6–7

## S

*SayMyName.py* program,  
 42–44, 44f  
*SayOurNames.py* program,  
 63–65, 64f  
 scale() function, 221  
 score  
 adding points, 240–241  
 displaying in game,  
 241–245  
 subtracting lives,  
 236–237  
 screenshots, 171  
 self parameter, 219–221  
 set\_caption() function, 208  
 setheading() function, 70–71

- set\_mode() function, 178, 192, 197
  - setpos() function, 112, 148–150, 158–160, 163, 166
  - setx() function, 70
  - sety() function, 70–71
  - shell, 5f, 269, 275
    - defined, 5, 298
    - doing math in, 36, 36f
    - syntax errors, 37–38, 37f
    - variables in, 38–39, 38f
  - ShowDot.py* program, 177–180, 177f, 203
  - ShowPic.py* program, 181–185, 181f
  - sides variable, 25, 26f, 27–28, 138
  - single quotation marks ('), 80
  - size variable, 115–116
  - SmileyBounce1.py* program, 190–197, 191f, 192f
  - SmileyBounce2.py* program, 197–200, 201f, 233–234
  - Smiley class, 218–219
  - SmileyExplosion.py* program, 215–224, 215f, 229
  - SmileyMove.py* program, 186–190
  - SmileyPong1.py* program, 233–245, 234f, 236f, 238f, 241f
  - SmileyPong2.py* program, 245–252, 249f, 261–262
  - SmileyPong3.py* program, 262
  - SmileyPopHitCounter.py* program, 262
  - SmileyPop.py* program, 224–227
  - SmileyPop2.py* program, 252–259
  - SmileyThrow.py* program, 229
  - sorting, defined, 298
  - sort() function, 127, 131
  - sound, adding with Pygame, 252–254
  - speed, 194–199
  - speed() function, 135
  - SpiralFamily.py* program, 65–67, 67f, 75
  - spiral() function, 165
  - SpiralMyName.py* program, 44–45, 46f, 52
  - SpiralRosettes.py* program, 74, 74f
  - spot variable, 210
  - Sprite class, 216–218
  - sprites
    - defined, 215
    - removing, 224–225
    - scaling, 221
    - setting up, 218–220
    - updating, 220–221
  - square brackets ([ ]), 46
  - square spirals
    - adjusted, 16–17
    - basic, 12–15
  - SquareSpiral1.py* program, 12–15, 12f
  - SquareSpiral2.py* program, 16–17, 16f
  - SquareSpiral3.py* program, 19–20, 19f
  - statements
    - defined, 37
    - syntax errors, 37–38
  - str() function, 242, 255
  - strings, 32, 42–44, 96–97
    - defined, 20, 42, 298
  - subtraction operator (-), 35, 35t
  - SuperSpiral.py* program, 292–293, 293f
  - surfaces, 178
  - symmetric ciphers (symmetric codes), 95
  - symmetry, 95
  - syntax
    - defined, 37, 298
    - errors, 37–38, 37f
- T**
- textinput() function, 44–45, 44f
  - text variable, 242
  - ThankYou.py* program, 33, 34f
  - tick() method, 190, 216
  - timer variable, 189
  - transformation, defined, 221
  - transparency, defined, 299
  - true division, 39
  - True value, 83–84
  - TurtleDrawMax.py* program, 159–160, 160f
  - TurtleDraw.py* program, 158–159, 159f, 163, 165
  - turtle graphics, 11–29
    - circles, 17–19
    - colors, 19
    - changing
      - background, 23
      - using multiple, 20–22
    - defined, 11–12
    - multi-sided spirals, 25–26
    - Pygame vs., 180–181
    - setting random
      - positions, 112
    - square spirals
      - adjusted, 16–17
      - basic, 12–15
      - website, 20
  - turtle module, 12–13.
    - See also* drawing; functions; loops; randomness
  - circle() function, 147

turtle module, *continued*  
forward() function,  
142–143  
listen() function, 162  
numinput() function, 47,  
59–60, 69  
onkeypress() function, 161  
onclick() function,  
158, 160, 163–166,  
168–170  
pendown() function, 112  
penup() function, 112  
setpos() function, 112  
textinput() function,  
44–45, 66  
window\_height() function,  
113–114  
window\_width() function,  
113–114  
write() function, 44  
turtlesize() function, 162  
typefaces (fonts), 242–243

## U

update() function, 179, 184,  
213, 220–221, 223,  
235, 255  
up() function, 161  
upper() function, 96, 99

## V

values, 32  
variables, 32–34. *See also*  
*names of specific*  
*variables*  
assigning values to, 32  
defined, 14, 32, 298  
naming, 32–33  
order of operations, 39  
in shell, 38–39, 38f  
true division, 39  
*ViralFamilySpiral.py*  
program, 75  
*ViralSpiral.py* program,  
68–72, 71f, 74–75

## W

War-style card game,  
119–125  
building deck of cards,  
119–120  
continuing play, 123–125  
counting cards, 121–123  
dealing cards, 120–121  
weight\_lb variable, 156  
*WhatsMyGrade.py* program,  
91–92  
*WhatToWear.py* program, 94  
while loops, 62–64, 65–66,  
78, 108, 123–125,  
130, 179, 214, 235  
defined, 299  
width() function, 25,  
150, 159  
window\_height() function,  
113–114, 135, 139  
window\_width() function,  
113–114, 135, 139  
write() function, 44

## X

x-axis, 70, 111, 180  
x variable, 14, 14n2, 38–39  
xvel parameter, 219–221

## Y

Yahtzee-style game,  
126–132  
probabilities, 131–132  
setting up, 126–127  
sorting dice, 127–128  
testing dice, 128–129  
y-axis, 70–71, 111, 180  
your\_age variable, 84  
your\_face variable, 121, 123  
*YourName.py* program,  
6–7, 7f  
your\_name variable, 33, 46  
your\_suit variable, 121  
yvel parameter, 219–221