

Index

SYMBOLS

- 382, 545
 - `__debug__` property
 - setting to False 502
 - `__eq__()` special method 544
 - `__hash__()` function 436
 - `__init__()` 501, 537
 - example 496
 - purpose of 496
 - `__main__.py` 203
 - `__name__`
 - module property 104
 - `__new__()`
 - example 496
 - purpose of 496
 - when to implement 497
 - `__new__()` and `__init__()`
 - calling order 497
 - `__pycache__` directory 199
 - `__str__()`
 - example 496
 - purpose of 496
 - `!=` 382
 - `.bash_aliases` file 48
 - `.bash_profile`
 - adding aliases to 46
 - `.bash_profile` file
 - purpose of 48
 - `.bashrc` file
 - purpose of 48
 - `.bat` files 18
 - `.dmg` images 38
 - `.gitignore` 290
 - configuring 291
 - file macOS users should add to 271
 - templates 291
 - `.gitignore` file
 - purpose of 129
 - `.gitignore` templates 129
 - `.ps1` files 18
 - `.py` file suffix 90
 - `.rpm` packages 38
 - `.ssh` directory
 - purpose of 254
 - `.venv` directory 313
 - purpose of 315
 - `.vimrc` file 32
 - 'f' string 362
 - Pipfile
 - 315
 - `@abstractmethod` decorator 525
 - `@property` 501
 - /etc/hosts file location 601
 - \ escape character 89
 - + operator 87, 89
 - <> 382
 - = 382
 - = equality operator 104
 - = operator 543
 - > 382
 - >= 382
 - `$_` Bash Prompt Generator 49
- ## A
- ABC class 525
 - abc module 525
 - abstract base class
 - definition 524
 - addition operator 87
 - applying to mixed types 89
 - age
 - calculate 106
 - Agile Software Development 116
 - algorithm 138
 - runtime growth rate 195
 - understanding the concept of 178
 - working definition of 189
 - algorithms 178
 - dominate time factor 192
 - good vs. better 190
 - alias
 - to Terminal application 10
 - analysis 121
 - analyst 66
 - and 383
 - Apache2
 - Ubuntu
 - installation 638
 - Apache2 Web Server 619
 - AppData folder
 - showing 5
 - Apple App Store 38
 - Apple MacBook Pro® 178
 - application
 - graceful recovery 119
 - Application Aliases
 - macOS 10
 - application architecture
 - multilayered 651
 - application configuration file 675
 - application distribution 561
 - application entry point 652
 - application framework
 - for common app services 675
 - application layers
 - logical 563
 - application logging 678
 - application security 675
 - application settings 677
 - ApplicationBase class 680
 - apt
 - installing python with 41
 - installing Terminator 27
 - apt package manager 41
 - architect 67
 - architecture design goals
 - easy to evolve 505
 - easy to maintain 505
 - easy to understand 505
 - argparse 586
 - argparse module 170, 585
 - ArgumentParser 170
 - setting properties 587
 - ArgumentParser class 586
 - array processing 119
 - ASCII 16, 467
 - assembler 183
 - assembly language 183
 - assertions 769
 - Association of Computing Machinery (ACM) 223
 - asymptotic runtime 195
 - attribute candidates 120
 - attributes 118
 - class scope 499

- object scope 499
- vs properties 500

AUTO_INCREMENT 658

B

- background thread 593
- base class 521
- base type 521
- baseline directory structure 288
- baseline project organization structure 288
- bash
 - default shell
 - change to 242
- bash build script 332–343
- bash command alias 46
- Bash script
 - cross-platform shebang 339
- bash script
 - case statement 340
 - command 341
 - comments 339
 - constants 339
 - if statement 341
 - make executable 337
 - read only variables 339
 - running 337
 - shebang 338
 - variables 339
- bash scripts
 - process command-line arguments 340
- bash shell 4
 - default launch mode
 - macOS 51
 - standardizing on 18
- basic system configuration tasks 4
- basic.vim file 32
- batch files 18
- behavior specification 524
- Bertrand Meyer’s Design by Contract 521
- big endian 478
- binary
 - image metadata 472
- binary data
 - reading and writing files 467
- binary strings vs. text 470
- binding to all host IP addresses 585
- bit 185
- blocking I/O 579
- Born Again Shell (bash) 17
- branch naming convention 251
- branching 380
- branching & looping statements

- quick reference guide 401

Branching Statements 384

breakpoints

- debugger 103

brew 40

- commands
 - list 312
- installing tree command 40

bubble sort 190

Bucket Approach x

build.sh 332

- baseline build script 332
- process command-line arguments 340
- rationale for 332
- running 337

build.sh Bash script 778

built-in input() 138

byte 185, 186

byte order marker (BOM) 478

bytearray 468

bytecode 187

bytes 468

C

- cache memory 184
- cache miss 185
- calling base class `__init__()` method from derived class 522
- cascade delete 654
- change management 249
- characters
 - print repeated 92
- child class 521
- chmod 337
- Cinnamon desktop 13
- CISC 182
- class
 - code layout 501
 - definition example 495
 - definition of the term 494
 - keyword 494, 495
 - what is a 494
- class diagrams 506
- class scope 499
- classes 494–515
 - defining and using
 - rules to preserve your sanity 498
- class-wide attributes
 - common rookie mistake 499
 - naive mistakes 500
- clear 164
- client
 - command-line arguments 585
 - definition 574
 - hardware & software 560
 - process server JSON response 607
- client-server architecture
 - definition 574
- client-server programming 574–610
- close() 457
- cls 164
- cmd.exe 46
 - running in Git Bash 46
- code
 - launch VS Code 36
- code coverage
 - unit tests 769
- code object file 199
 - command-line execution 199
- Code Smell 144
- coding mistakes
 - common 91
- cohesion 355
- columns 654
- command aliases
 - create in Linux 48
 - in Git Bash 46
 - macOS 51
- Command mode
 - Vim 32
- Command not found error
 - what to do 45
- Command Prompt 6, 7, 16, 18
- command-line interfaces
 - definition 16
- command-line parameters 170
- comments
 - using to define algorithm 138
- Comparison Operators 382
- Complex Instruction Set Computer 182
- computer
 - architecture
 - feature set 182
 - feature set accessibility 182
 - feature set implementation 182
 - three aspects of 182
 - definition of 178
 - developer friendly configuration 4
 - memory
 - organization 184
 - system 178
 - components of 179
 - keyboard 179
 - memory 184
 - monitor 179
 - mouse 179
 - speakers 179
 - system unit 179

- vs. computer system 178
- what makes one an amazing device 178
- Computer Management 7
- computer network
 - definition 556
- computers 178
 - why they are powerful 188
- concatenation
 - need for str() function 414
 - string 87
 - results of 87
- Concept Of The Flow 74
- conceptual complexity
 - taming 494
- concrete data type 528
- Conditional Expressions 380
- conditional expressions 380
- configuration checklist 54
- configuration management 248
- constant 120
- constructor
 - calling 497
- context.py file
 - purpose of 299
- control flow statements 380–403
- Control Panel 7
- cooling system 180
- Core ML 183
- Core ML Tools 183
- coupling 356
- create_database.sql 723
- create_tables.sql 724
- ctrl-c
 - purpose of 584
- ctrl-c vs. ctrl-z 584
- ctrl-z
 - purpose of 584
- current position 121
- customize bash prompt
 - Linux 49
 - macOS 51, 269

D

- daemon thread 593
- Data Control Language 655
- Data Control Language (DCL) 650
- Data Definition Language 655
- Data Definition Language (DDL) 650
- Data Manipulation Language 655, 659
- Data Manipulation Language (DML) 650

- data type 120
- database 650
 - cascade delete 654
 - columns 654
 - foreign key 654
 - primary key 654
 - referential integrity 654
 - rows 654
 - scripts 657
 - table 654
 - tables
 - create with script 657
 - verify creation 656
- database connection
 - python 650
- database connection pools 686
- database connections
 - troubleshooting 688
- Database First Approach 704
 - explained 704
- database management system 618, 653
- database query parameters 688
- database row tuple
 - example 688
- database scripts
 - create individual scripts 721
- database user
 - creating 683
 - script 684
- date 501
- date calculations 106
- datetime 501
- datetime module
 - usage 385
- datetime.isoformat() 447
- DBMS 618, 653
- DDL 655
- Debian 13
- Debian package manager 38
- debug session 103
- debugger
 - breakpoint 102
 - purpose of 102
- def keyword 104
- Default String Representation Behavior 537
- defining a class 495
- derived class 520, 521
- derived type 521
- deserialization
 - definition 481
- design
 - how to arrive at good 505
- design project-approach strategy area

- applied 123
- deterministic builds 320
- developer's agreement 145
- development cycle
 - applying 116
 - code 115, 804
 - demo of 94
 - plan 115, 804
 - refactor 116
 - test 115, 804
 - using 116
- development environment 4
 - baseline 4
- development projects folder
 - create
 - Linux 14
 - macOS 13
 - Windows 9
- development sprints 116
- development team
 - Musketeer's agreement 252
- DevOps 232
- Pipfile
 - 315
- dict() constructor 438
- dictionaries 436–451
 - acceptable key types 437
 - converting to JSON 444
 - create and populating 437
 - datetime string format 447
 - general behavior with duplicate keys 438
 - random seeding of hash values 438
 - requirements for tuple keys 437
 - rules for creating to produce valid JSON 445
 - supported operations 441
 - use cases 448
 - using sorted() function to sort keys 444
- dictionary 436
 - conceptual view 436
 - definition 436
 - keys
 - need for immutability 436
 - view objects 442
- dictionary comprehension 440
- dictionary comprehensions 439
- dictionary key
 - hash values
 - behavior 437
- dictionary view objects 443
- direction 121
- directory 4
- dis module 202
- disassembling modules 202
- DML 659

- doc comment 104
 - module level example 385
- doc comments 501
- docs directory 290
- dpkg
 - installing VS Code 35
- drop_database.sql 722
- drop_tables.sql 723
- drop_user.sql 722
- dunder methods 536
- Dynamic Host Configuration Protocol (DHCP) 580
- dynamic modeling 504

E

- echo client
 - implementation 581
- echo client entry point 582
- emoji
 - CLDR short names 411
- Employee Training App
 - Add New Employee 748
 - Infrastructure Layer 734
 - listing employees 743
 - Presentation Layer 740
 - Service Layer 731
- endianness 478
- Engineer's Notebook 253
 - Importance of 618
- entity 504
- Enum class 156
- enumerate() function 422
- enumerate() method 163
- enumerated types 156
- environment variable
 - definition 44
 - PATH 306
- environment variables 4
 - export
 - macOS 52
 - export in Linux 50
 - setting in Windows 45
 - System
 - don't dork up! 44
 - System vs. User 44
- environment variables and shell profiles 44–53
- Environment Variables dialog 45
- ephemeral port 587
- EPROM 183
- Erasable Programmable Read-Only Memory 183
- error checking 119
- escape character 89
- escape sequence 89, 388

- escaped characters 135
- escaped tab character 135
- evolving the database 756
- except clause 104
- Exception 94
 - object 104
- exception
 - runtime 94
- exceptions 93
 - catching multiple 457
 - code that can throw 104
 - raising 369
- Exchangeable Image File Format (EXIF) 472
- EXIF 472
 - tag record structure 473
- exponential runtime 195
- expression 87

F

- f.read() 457
- falsy 380
- file
 - unstage after git add 272
- File Explorer
 - Quick access view 5
 - Windows 5
- File I/O 456–486
 - Basic File Operations 456
- file I/O
 - binary 467
 - random 467
 - seeking 469
- file mode 457
- file modes
 - binary 459
 - table 458
 - text 458
- file object 459
 - methods 459
- file object alias 457
- file paths 461
 - absolute 463
 - os-agnostic 464
 - relative 461
 - separator
 - windows vs. unix 463
 - Windows 464
- file pointer 458, 468, 470
- file suffixes
 - show
 - macOS 10
- files and folders
 - hidden
 - show
 - Linux 14
 - macOS 9, 10
 - show hidden
 - Windows 5
- Finder window
 - open in home directory 13
- Firestorm® P-cores 182
- Fixing Common Errors 91
- floor 121
- flow 64, 66
- folder 4
- folder vs. directory
 - term interchangeability 4
- for loops
 - nested 148
- for statement 393
 - else clause 396
- foreign key 650, 654
- foreign key constraint 654
- formatted string 362
- formatted strings 104
- formatting
 - source code 804
- function
 - client view 354
 - developer view 354
 - viewed as a black box 354
- functions 350
 - 10,000 foot view 354
 - anatomy of 356
 - assign to variables 371
 - built-in
 - float() 89
 - input() 92
 - print() 88
 - str() 89
 - calling from external module 359
 - calling within same module 358
 - checking argument's type 368
 - defining and calling 356
 - design considerations 355
 - difference between definition and call 105
 - first class objects 371
 - minimize side effects 356
 - naming 357
 - optional arguments 367
 - parameters
 - keyword-only 365
 - positional-only 364
 - positional-or-keyword 363
 - var-keyword 366
 - var-positional 365
 - parameters and arguments 362
 - pass as arguments to 371
 - purpose of 355
 - python built-in 354

- return data from 370
- type hints 367
- use verbs in name 358
- with leading underscore 358

fundamental language features 119

G

Git 18, 125, 249

- branching 252
- branching workflow 252
- branching workflow command summary 280
- checking version 233
- clone
 - what happens 251
- description 250
- Linux 234
- Linux Mint installed version 234
- list local branches 275
- macOS 233
- pull request
 - purpose of 278
- save changes to local repository 272
- simple workflow 252
- stage files for next commit 271
- switch between branches 275
- typical team workflow 251
- unstage file after git add 272
- update with brew 233
- verification 232
- Windows 232

Git Bash

- customize bash prompt 47
- default terminal 20
- mintty 20

Git Bash (mintty) 44

Git Bash (mintty) terminal 232

Git Bash Terminal

- properties 20

Git Bash terminal

- option groups 20
- running python first time 42
- Vim running in 30

Git Branching and merging 274

Git commands

- add 271
- branch 275
- checkout 275
- clone 270
- commit 272
- push 272
- simple workflow command summary 273
- status 270

Git For Windows 232

Git for Windows 18

- installation 18
- Vim 30

git repository

- verify operations 130

Git SCM 18

Git-Bash

- VS Code embedded terminal 101

GitHub 125, 249

- add public SSH key to 257
- copy code from 220
- create account 235
- multi-factor authentication 235

good software design

- hallmark of 520

Google Dorks 213, 222

Google Trends 213

GPU 182

Graphics Processing Unit 182

Guest folder

- macOS 12

H

hash value 436

hash value random number salting 436

hash values

- random number seeding 438

Hello World!

- example program 90

heterogeneous networks 557

hidden files and folders

- macOS
 - show 12

high-level software-architecture diagram 123

High-Value Online Help Resources 215

home directory name

- no spaces in 4

Homebrew 40

- installing 40

homogeneous networks 557

hosts file contents 601

I

IANA Service Name and Transport Protocol Port Number Registry 580

Icestorm® E-cores 182

Identity Operators 383

If Statement 384

if statement 104

- purpose of 384

if/elif/else statement 389

If/Else Statement 386

ifaddr

- get all host IPv4 IPs 592

ifaddr package 592

immutable sequence 408

Implementation Approach 124

in 383

in operator 415

incidental system administration 4

inheritance 520–532

- behavior specification 524
- commonly-used vocabulary 521
- extending behavior 520
- is-a relationship 521

initial housekeeping

- macOS 9
- purpose of 15
- Windows 5

initialize_database.sh 729

inner classes 156

input() function 138

Installation Checklist

- need for 619

instance 495

instance attributes 496

instantiation call 133

instantiation operation 133

Instruction Set Architecture 183

Integrated Development Environment (IDE) 35

intelligent terminal 17

interfaces

- the role of 528

introspection

- to obtain executing method name 691

is 383

is not 383

is operator 542

ISA 183

is-a relationship 521

isinstance() 384

isinstance() function 529

isinstance() method

- usage 369

ISO 8601 447

issubclass() 384

iterable sequence 393

iterating 380

iteration 393

- development 116

iterative development 116

iterator 380

- list processing 415

iTerm2 22

- Auto-Rainbow 26

- Configure Status Bar 26
 - installation 22
 - preferences 25
- Shell Integration 24
- shell integration
 - testing 27
- status bar
 - enabling 26
- Terminal Configuration 24
- iTerm2 Shell Integration
- Marks 27

J

- JavaScript Object Notation (JSON) 408, 426
- JIRA 251
- JPEG
 - End-of-Image (EOI) 472
 - Start-of-Image (SOI) 472
- JSON
 - saving to file 483
 - server response structure 595
 - write to file with open() 447
- JSON configuration file 675
 - reading from command line 676
- json.dumps() 426, 447
- json.loads() 426, 447

K

- key 436
- key/value pairs 436
- keys
 - immutable 436
- keywords
 - def 104
 - with 457

L

- language features 115, 803
- Language Features Strategy Area
 - purpose 122
- leading underscore
 - purpose 501
- Learned Helplessness 213
- len() 408
- linear time 195
- Linux
 - baseline configuration 13
 - configure for software development 4
 - customize bash profile 48
 - terminal
 - Terminator 27
 - Trash icon

- add to desktop 14
- Linux commands
 - cd 22, 30
 - mkdir 30
- linux commands
 - chmod 337
 - popd 300
 - pushd 300
- Linux Mint 13
 - Cinnamon Desktop 13
- Liskov Substitution Principle (LSP) 521
- list
 - convert to JSON 426
 - customized sort 425
 - definition 419
 - extracting index and value from iterable 163
 - processing 421
- list comprehension 148
- list comprehensions 420
- list processing 119
 - iterator 415
- lists 419
 - creating and initializing 419
- literal
 - numeric 87
- literals
 - string 87
- little endian 478
- local area network 556
- localhost address 580
- logging convenience class 678
- Logical Operators 383
- login shell 48
- login shell vs. non-login shell 48
- looping 380
- looping statements 393

M

- M1 Max processor 178
- M1 Max SoC 181
- machine code 183
- machine instructions 183
- macOS
 - compatible OS and Xcode versions 23
 - configure for software development 4
 - desktop and dock
 - add aliases to 10
 - Finder window
 - change layout 11
 - Guest folder 12
 - hidden files and folders 10

- iTerm2 22
- login shell default behavior 51
- Monterey 24
- package manager
 - brew 40
- python vs python3 43
- Sequoia 24
- Sonoma 24
- Tahoe 24
- Terminal 10
- Ventura 24
- main thread 575
- main.py module
 - purpose of 124, 296
- make alias
 - macOS 10
- MAMP 618
 - installation
 - endgame 618
 - installation process
 - overview 619
 - macOS 618
 - installation 630
 - Windows 618
 - installation 620
- MAMP PRO 618
- Markdown 128
- match statement 143, 391
 - default case 143
- match statements
 - nested 156
- max() 408
- Membership Operators 383
- memory
 - bit 185, 186
 - byte 185, 186
 - non-volatile 184
 - organization 184
 - RAM 184
 - ROM 184
 - volatile 184
 - word 185, 186
- memory hierarchy 184
- mental blockers
 - categories 212
- menu 120
- merge sort 193
- message digest 482
- Metal Shading Language 183
- method
 - definition 509
- method input
 - validation 787
- method stubbing 137
- method stubs 125
- methods 119, 509
 - defining and calling 510

- methods vs. functions 509
- microcode 183
- min() 408
- mindfulness 291
- minty 20
- mistakes
 - coding
 - common 91
 - common
 - forgetting to save file between edits 94
- model 119
- modeling 118
- module
 - explicit entry point 353
- modules 90, 350
 - `__all__` property 360
 - choosing appropriate names for 352
 - disassembling 202
 - how they execute 105
 - importing 106
 - exclude internal functions 359
 - purpose and use 350
 - responsibilities 352
 - valid names 351
- modules and functions 350–373
- most important configuration 4
- MS-DOS commands
 - aliases for 47
- multihomed
 - definition 585
- multilayered applications 563
- multilayered architecture 651
- multithreaded server 589
 - multiple clients connecting to 589
- MultiThreadedEchoServer
 - implementation 590
- mutable sequence 408
- MySQL 618, 619
 - Ubuntu
 - installation 638
- mysql
 - client directory 619
- MySQL Server
 - default port
 - macOS 619
 - Windows 619
- MySQL statement
 - importance of semicolon 660
- `mysql_persistence_wrapper.py` 680
- `mysql-connector-python` 650
 - installation 652

N

- nested for loops 148
- nested function call
 - example of 370
- network
 - purpose of 556
 - types 557
- Network applications 556
- network program
 - cross-platform testing 580
- network protocols
 - role of 557
- networking
 - fundamentals 556–570
- networking protocols
 - Internet 565
- Neural Engine 182
- Neural Processing Unit 182
- non-login shell 48
- non-volatile memory 184
- Normal mode
 - Vim 30
- not 383
- not in 383
- noun 120
- noun lists
 - suggesting possible application objects 119
- nouns 119, 121
 - mapping to data structures 120
- Noun-Verb Analysis 119
- NPU 182
- numeric literals 87
- numeric types 87

o

- O switch 197
- object
 - definition 495
- object attributes 119
- object comparisons
 - two types of 380
- object equality 543
- object identity 542
- object instantiation 495
- object scope 499
- Object String Representation 537
- object usage checklist 549
- Object-Oriented Analysis 504
- object-oriented analysis & design 494
- Object-Oriented Analysis, Design, And Programming 503
- Object-Oriented Design 504

- object-oriented first approach 114
- Object-Oriented Programming 504
- object-oriented programming 494
- objects
 - creating from class 496
 - initialize into known state 496
 - natural ordering 545
- ODBC@localhost 619
- OOA&D
 - What's the point? 505
- open() 447, 457
- operating system 187
- operating system check 578
- operator precedence 386
- operators
 - addition 87
- or 383
- organization
 - source code
 - need for 97
- os 578
- os module 169
 - environ dictionary 449
- `os.getcwd()` 465
- `os.makedirs()` 465
- `os.path.join()` 464, 465

P

- package
 - python-dateutil 107
- package manager
 - purpose of 37
- Package Managers 37, 37–41
- package managers
 - apt 27, 38
 - Homebrew 40
 - rpm 38
 - winget 39
- packages
 - installing 106
 - PrettyTable 422
 - prettytable 381
 - third-party 106
- Pipfile
 - 315
- parent class 521
- parentheses
 - to enforce operator precedence 386
- Part VI Preliminaries 618–647
- PATH 306
- PATH variable 44
- paths
 - relative vs. absolute 461
- Peer Review 274
- peer review 249

- pen 121
- PEP 8
 - guidance on constants 145
- Persistence Layer
 - responsibilities 651
- persistence layer 680
- PHP 619
- PHP & Apache PHP Module
 - Ubuntu
 - installation 639
- PhpMyAdmin 655
- phpMyAdmin 618, 619
 - create cross-reference table 709
 - create database 705
 - create database user 713
 - create tables 706
 - defining foreign key relationships 711
 - export database SQL script 715
 - export database user script 720
 - Ubuntu
 - installation 640
- physical complexity
 - taming 494
- pickle 481
- Pipenv 306, 311
 - create virtual environment 311
 - run application with 313
 - sufficient reasons to use 312
- pipenv
 - commands
 - check 323
 - graph 323
 - run 314
 - run vs. shell 314
 - shell 314
 - uninstall 322
 - deterministic builds 320
 - how to use 306–327
 - initialize virtual environment 652
 - install packages for development 318
 - install packages for production 321
 - install specific package version 318
 - running package security scans 324
 - testing
 - PIPENV_VENV_IN_PROJECT 239
- pipenv run 314
- pipenv subshell
 - exit 315
- PIPENV_VENV_IN_PROJECT 311, 313
- Pipfile 306, 315
 - purpose of 315
 - sections 315
- Pipfile.lock 306
 - deterministic builds 320
- polynomial time 193
- popd command 300
- port number 580
- PowerShell 7, 16, 18, 41
 - cmdlets 18
- PowerShell scripts 18
- Practice breeds confidence! 114
- prepared statements 688
- Preparing for database programming 618–647
- Presentation Layer
 - responsibilities 651
- PrettyTable 422
- primary key 650, 654
- problem domain 115, 803
 - strategy area
 - applied 119
- processes and threads 575
- processing cycle
 - decode 189
 - execute 189
 - fetch 188
 - store 189
- processor
 - machine code 183
- productivity dividend 768
- professional software engineer
 - if you intend to be 248
- program
 - computer perspective 187
 - definition 187
 - human perspective 187
- programmer 67
- programs 178
 - run from command line 90
 - why they crash 189
- Project Approach Strategy x
- project approach strategy 66
- project folder 125
 - think about first 97
- project objectives 118
- Project Organization 288–303
- project repository 125
- Project Specification
 - typical classroom 117
- project specification 121
- project structure
 - .gitignore 290
 - docs directory 290
 - multilayered application architecture 651
 - README.md 290
 - src directory 289
 - tests directory 289
- project-approach strategy 68, 114, 115
 - applying 118
 - areas of concern 68
 - high-level design and implementation strategy 69
- properties 510
 - vs. attributes 500
- protocol
 - definition 559
- Pull Request 274
- pull request 251, 276
 - purpose of 278
- pushd command 300
- Pyenv 323
- pytest 768
 - installing 770
- pytest-cov plugin 776
- Python
 - command-line module execution 198
 - command-line script execution 197
 - execution process 196
 - how programs run 196
 - interactive interpreter 86
 - interactive mode 86
 - listing installed versions 311
 - loading and parsing program 94
 - package conflicts
 - explained 308
 - pipe script file to interpreter 198
 - REPL 86
 - using 87–90
 - running
 - Linux 86
 - macOS 86
 - Windows 86
- python
 - install and configure 41–44
 - install on Linux 43
- Python database programming 683
- Python docs
 - navigating 214
- Python Interpreter
 - exiting 89
- Python interpreter 90
- python interpreter 41
- python -O switch 503
- Python Package Index (PyPI) 108
- Python process 575
- Python Standard Library 108
- python vs python3
 - macOS 43
- python3
 - install with brew 42

Q

- quotation marks
 - single or double
 - use of 87
- quote marks
 - escaping within string 388

R

- random access memory (RAM) 184
- random file I/O
 - seeking 470
- random module 598
- range() function 395, 427
- ranges 427
- RDBMS 654
- README.md
 - documenting project with 292
- README.md File 128
 - purpose of 128
- README.md file
 - purpose of 290
- read-only memory (ROM) 184
- recursive call 195
- Reduced Instruction Set Computer 182
- referential integrity 654
- related tables 665
 - one-to-many 665
- relational database 650, 654
 - fundamentals 650–702
- relational database management system 654
- relationships
 - between database tables 654
- relative paths 461
- reliable software
 - designing 350
- REPL 41
- repr() 537
- requirements 115, 803
 - gaining insight through pictures 121
- Pipfile 315
- reserved port numbers 580
- resource sharing 556
- revision tracking 248
- RISC 182
- Robot Rat Application
 - Sprint 0 125
 - Sprint 1 132
 - Sprint 2 134
 - Sprint 3 137
 - Sprint 4 146

- Sprint 5 149
- Sprint 6 158
- robotrat_app.py module 132
- rows 654
- rpm 38
- run VS Code from command line 36

S

- sanity rules
 - One Class per Module 498
- SCM
 - Architecture And Processes 249
 - benefits 248
 - by different names 248
 - conflict resolution 249
 - defined 248
 - Git 250
 - Git workflows 250
 - primary architectural components 249
 - primary goal 248
 - remote repository 249
 - repository branches 249
- scripting the database 704–760
 - automating with bash script 729
- Scrum 116
 - process highlights 116
- section 315
- Secure Shell (SSH) 253
- seek() method
 - usage table 469
- self
 - purpose of method parameter 510
- sequence 408
 - definition 408
 - iteration
 - non-Pythonic 415
 - Pythonic 415
 - mutable vs. immutable 409
 - slicing 417
- sequence diagrams 506
- sequence types 408
- sequences 408–430
 - common operations 409
 - immutable 408
 - mutable 408
 - searching 415
- serialization
 - definition 481
- serializing objects
 - pickle 481
- server
 - application 560
 - creating JSON response 599
 - definition 574
 - executing client commands 596
 - hardware 560
 - processing client commands 599
- servers vs. clients 560
- Service Layer
 - responsibilities 651
- Services 7
- shell 16, 17
 - definition 17
- shortcuts
 - c drive
 - windows 8
 - creating
 - Windows 6
 - drive and folder
 - Windows 8
- signal handlers 584
- signal module 584
- Single vs. Double Quotes 388
- single-threaded echo server 575
 - implementation 577
- SingleThreadedEchoServer 578
- site-packages 307
- small victories 86
 - need for 213
 - purpose of 86
- SoC 181
- socket 578
 - setting socket options 578
- socket.accept() 579
- socket.AF_INET 578
- socket.bind() 578
- socket.listen() 578
- socket.SO_REUSEADDR 578
- socket.SOCK_STREAM 578
- software development
 - sprints 116
- software development cycle 114, 115, 116
 - employing 116
- software development roles 66
- software engineer
 - battle cry of 40
- Software Quality Assurance 768
- Software Quality Assurance (SQA)
 - defined 769
- sorted() function 547
- sorted() function vs. list.sort() 548
- source code
 - file header 804
 - formatting 804
 - improving readability 595
- Source Code Management (SCM) 248–286
- source code organization 97
- special methods 509, 536
 - __eq__() 544

- `__lt__()` 545
- species *Algorhynchus Logicus*
 - mating call of 541
- Sprint Zero 0 125
- sprints 116
- SQL 650, 655
 - commands
 - alter 655
 - create 655
 - drop 655
 - use 655
 - Data Control Language 655
 - Data Definition Language 655
 - Data Manipulation Language 655
 - three sub languages 655
- SQL Commands
 - ALTER TABLE 658
 - CREATE 656
 - DROP DATABASE 657
 - SHOW 656
- SQL commands
 - DELETE 663
 - DESCRIBE 659
 - INSERT 659
 - SELECT 660
 - UPDATE 662
 - USE 659
- SQL Data Types 659
- SQL functions
 - COUNT() 663
 - LAST_INSERT_ID() 673
 - LEFT() 663
- SQL Scripts
 - run from command line 657
- SQL scripts
 - comments 658
- SQL variables
 - setting 673
- src directory 289
- SSD flash memory 179
- SSH
 - passphrase 256
- SSH agent
 - starting 259
- SSH Key
 - generation and configuration 253–263
- SSH Keys
 - generation
 - command summary 263
 - video 253
- SSH keys 253
 - add private key to SSH agent 259
 - known_hosts file 260
 - process overview 254
 - ssh-keygen 255
- ssh-keygen 255

- Stack Overflow
 - etiquette 220
 - Expected Behavior 220
 - How To Ask A Question 220
- Stages of Flow 74
- state transition diagram 151
- State Transition Diagrams 149
- statements
 - match
 - default case 143
- static modeling 504
- `str.count()` 415
- `str.find()` 415, 416
- `str()` 414, 537
- string
 - concatenation 87
 - definition 410
 - literal 87
 - quote types
 - good practice 87
 - quotes
 - embedded 89
 - slicing 417
 - splitting 107
- string iteration 394
- `string.split()`
 - default delimiter 390
- strings 410
 - accessing individual characters 414
 - common operations 412
 - concatenation 413
 - embedded quotes 88
 - searching 415
 - single-quoted vs. double-quoted 87
- Structured Query Language 655
- Structured Query Language (SQL) 650
- stubbing 137
- subclass 521
- Sublime Text 34
 - set path to
 - macOS 52
- subshell
 - launch with pipenv 314
- subtype 521
- `super().__init__()` 522
- superclass 521
- supertype 521
- syntax errors
 - fixing 93
- sys 578
- system administration 4
- system board 179
- System Settings dialog 45
- system unit 179
- System vs. User environment vari-

- ables 44
- System-on-a-Chip 181

T

- table 650, 654
 - relationships
 - many-to-many 669
- tables
 - relationships
 - one-to-many 665
 - trouble with 669
- TCP/IP 557
- TCP/IP Protocol Stack 566
- technical debt 157, 158
- teletype 16
- teletype terminals 16
- Terminal 16
 - macOS 10
- terminal 16
 - minty 20
- terminal application 16
- Terminal Applications 15
- terminal applications
 - purpose 16
- terminal emulators 17
- terminal screen
 - clearing
 - different operating systems 165
 - clearing from Python code 164
- Terminal.app 10
- terminals
 - history 16
- Terminator 27
 - add to panel and desktop 27
- test fixtures 768
- tests directory 289
- text editors 29
 - Nano 33
 - Sublime Text 34
 - vi/vim 30
- The Art Of Programming 72
- The Processing Cycle 188
- The Software Development Cycle 75
- The Ultimate vimrc 32
- thread
 - background 593
 - definition 593
 - creating new to communicate with
 - client 591
 - daemon 593
 - setting target and args parameters 593
 - starting 593

- threads
 - benefit of additional 575
 - uses of 576
- tilde character
 - home directory 22
- Time to First Commit 248
- Transaction Control Language (TCL) 650, 655
- triple-quote strings 414
- truthy 380
- TruthyFalsy program 380
- try statement 104
- tty 16
- tuples 428
- tutor
 - need for 212
 - role of 213
- two-item tuple 163
- Type Checking 384
- type checking 529
- Type Conversion 89
- type conversion 103
- type hints 367
- types
 - numeric 87

U

- Ubuntu 13
 - package manager 38
- Ugly Baby 144
- UMA 181
- UML 506
 - class diagram 507
 - expressing inheritance 521
 - sequence diagram 508
- UML class diagram 124
- UML diagram
 - with abstract base classes 524
- Unicode 467, 470
 - Basic Latin 470
- Unicode (Basic Latin) 411
- Unicode code point 410
- Unicode Transformation Format 467
- Unified Memory Architecture 181
- Unified Modeling Language (UML) 124, 506
- unit test
 - assertions 769
- unit testing 768–796
 - prepare project for 770
- unit tests
 - add `__init__.py` file to tests directory 772
 - artifact naming guide 776

- as early-warning indicators 769
- code coverage 769
- context.py file 299, 771
- defined 769
- edge cases 784
- edge-case types 784
- install pytest package 770
- pytest
 - plugins 776
 - running with `-sv` flags 773
 - pytest-cov plugin 776
 - tests directory 771
 - writing and running 772
- unittest
 - running 300
- unstage file 272
- unterminated string literal 94
- use-case modeling 504
- user-defined type 536
- user-defined types 494, 536
- USS America (CV-66)
 - navigation station plaque 280
- UTF-8 467

V

- value 436
- variable 120
 - definition 88
- verb phrases 119
- verbs 119
- version control 248
- versioned database scripts 756
- vi 30
- Vi/Vim 30
- view objects
 - dictionary 442, 443
- Vim
 - as common denominator 30
 - basic usage 30
 - buffer 31
 - customize 32
 - four modes 30
 - modes
 - Command 30, 31
 - Insert 30, 31
 - Normal 30, 31
 - Visual 30, 31
 - Visual Block 31
 - Visual Line 31
 - run as administrator 47
 - set nu 32
 - show line numbers 32
 - turn on line numbers 32
- vim 30
 - Basic configuration 32
- virtual environment 306

- defined 306
- problems solved by using 310
- virtual environments 306–327
- virtualenv 306
- Visual Block
 - Vim 31
- Visual Line
 - Vim 31
- Visual Studio Code 35, 96
 - debugging programs 102
 - `--disable-gpu` 100
 - embedded terminal 99
 - fixing 100
 - Git Bash 101
 - installation 35
 - launch from command-line 36
 - Python extensions 35
 - running program 99
 - update 96
- Visual Studio JSON
 - Sort Document feature. 448
- volatile memory 184
- Von Neumann architecture 187
- VS Code
 - set path to
 - macOS 52
- VT100 17

W

- walking
 - helps you think 135
- Well-Behaved Objects 536–550
- while statement 397
 - else clause 399
- Windows
 - configure for software development 4
 - environment variables 45
 - host file location 601
- Windows command interpreter 46
- Windows commands
 - running in Git Bash 46
- Windows subsystem for Linux 18
- winget 38, 39, 41
 - installing python with 39
 - using 39
- winpty
 - run python with in Git Bash 42
 - running python with 47
- winpty command 42
 - purpose of 42
- with 457
 - context manager 457
- word 185, 186
- working directory
 - defined 461

WSL 18
Windows Subsystem for Linux 18

X

Xcode 23
 command-line tools 24
Xcode and Command-Line Developer Tools
 installing 23
Xcode Releases
 website 24
xterm 20

Y

You're Stuck — Now What? 212
yum 38

Z

Z Shell (zsh) 17