

Contents

Fixtures and Mocking in Python	
How do you test Moon-landing?	
How do you test a system	
Plan	
About me	
Goal	
Fixtures	
Fixtuers in Pytest	
Traditional xUnit fixtures	
Dependency Injection	
Temporary directory - tmpdir	
Capture STDOUT and STDERR - capsys	
Home-made fixture	
Home-made fixture - conftest	
Home-made fixture with tmpdir	
Home-made fixture with yield	
Fixture Autouse	
Fixture Autouse with yield	
Fixture for MongoDB	
Test Doubles	
Test Doubles explained	
Verify behavior or state?	1
What is Mocking and Monkey Patching?	1
Situations	1
Unit testing vs. Integration testing	1
Experiment with mocking in various situations	1
Examples are simple	1
Hard coded path	1
Manually Patching attribute	12
Monkey Patching attribute	13
Monkey Patching functions	13
Monkey Patching dictionary items	14
Mocking a whole class	15

CONTENTS

Mocking input/output	15
Mocking input/output	16
Mocking random numbers	17
Exercises	17
Work in pairs	17
Exercise: test login expiration	17
Solution: test login expiration	18
Exercise: Record e-mail sending	19
Solution: Record e-mail sending	20
Exercise: Fixture database	21
Exercise: One Dimentional space-fight	21
Exercise: web client	23
Exercise: Open WeatherMap client	23
Exercise: Mocking A Bank	24
Testing the whole application	26
Resources	27
Retrospective	27
Job searching help	27
Solutions - game	27
Solutions - Mocking the database access	29
First steps	31
What is Python?	31
What is needed to write a program?	31
The source (code) of Python	32
Python 2 vs. Python 3	32
Installation	32
Installation on Linux	33
Installation on Apple Mac OSX	33
Installation on MS Windows	33
Editors, IDEs	34
Documentation	36
Program types	36
Python on the command line	36
First script - hello world	37
Examples	37
Comments	37
Variables	38
Exercise: Hello world	38
What is programming?	38
What are the programming languages	38
A written human language	39
A programming language	39

Words and punctuation matter!	39
Literals, Value Types in Python	39
Floating point limitation	40
Value Types in Numpy	40
Rectangle (numerical operations)	40
Multiply string	40
Add numbers	40
Add strings	40
Exercise: Calculations	40
Solution: Calculations	40

Second steps	40
Modules	40
A main function	40
The main function - called	40
Indentation	40
Conditional main	40
Input - Output I/O	40
print in Python 2	5
print in Python 3	5
print in Python 2 as if it was Python 3	5
Exception: SyntaxError: Missing parentheses in call	5
Prompting for user input in Python 2	5
Prompting for user input in Python 3	5
Python2 input or raw_input?	5
Prompting both Python 2 and Python 3	5
Add numbers entered by the user (oups)	5
Add numbers entered by the user (fixed)	5
How can I check if a string can be converted to a number?	5
Converting string to int	5
Converting float to int	5
Conditionals: if	5
Conditionals: if - else	5
Divide by 0	60
Conditionals: if - else (other example)	60
Conditionals: else if	61
Conditionals: elif	61
Ternary operator (Conditional Operator)	62
Case or Switch in Python	62
Exercise: Rectangle	62
Exercise: Calculator	62
Exercise: Age limit	63
Exercise: What is this language?	

CONTENTS

Exercise: Standard Input	63
Solution: Area of rectangle	63
Solution: Calculator	64
Solution: Calculator eval	65
Solution: Age limit	66
Solution: What is this language?	66
Command line arguments	67
Command line arguments - len	67
Command line arguments - exit	68
Exercise: Rectangle (argv)	68
Exercise: Calculator (argv)	68
Solution: Area of rectangle (argv)	68
Solution: Calculator (argv)	69
Solution: Calculator eval	70
Compilation vs. Interpretation	70
Is Python compiled or interpreted?	71
Flake8 checking	72
Pylint checking	72
Numbers	74
Numbers	74
Operators for Numbers	74
Integer division and the future	75
Pseudo Random Number (uniform distribution)	76
Fixed random numbers	76
Rolling dice - randrange	77
Random choice	77
built-in method	77
Exception: TypeError: 'module' object is not callable	78
Fixing the previous code	78
Exception: AttributeError: module 'random' has no attribute	79
Exercise: Number guessing game - level 0	80
Exercise: Fruit salad	80
Solution: Number guessing game - level 0	80
Solution: Fruit salad	80
Comparison and Boolean	82
if statement again	82
Comparison operators	82
Compare numbers, compare strings	82
Do NOT Compare different types!	83
Complex if statement with boolean operators	84
Boolean operators	85

Boolean truth tables	85
Boolean values: True and False	86
Using True and False in variables	86
Comparison returns True or False	87
Assign comparisons to variables	87
Flag	88
Toggle	88
Short circuit	89
Short circuit fixed	89
Does this value count as True or False?	90
True and False values in Python	91
Incorrect use of conditions	9
Exercise: compare numbers	9
Exercise: compare strings	9
Solution: compare numbers	9
Solution: compare strings	9

strings	9
Single quoted and double quoted strings	9
Long lines	9
Triple quoted strings (multiline)	9
String length (len)	9
String repetition and concatenation	9
A character in a string	9
String slice (instead of substr)	9
Change a string	9
How to change a string	9
String copy	9
String functions and methods (len, upper, lower)	10
index in string	10
index in string with range	10
rindex in string with range	10
find in string	10
Find all in the string	10
in string	10
index if in string	10
Encodings: ASCII, Windows-1255, Unicode	10
raw strings	10
ord	10
ord in a file	10
chr - number to character	10
Exercise: one string in another string	10
Exercise: to ASCII CLI	10

Exercise: from ASCII CLI	107
Solution: one string in another string	107
Solution: compare strings	107
Solution: to ASCII CLI	108
Solution: from ASCII CLI	108
Loops	
Loops: for-in and while	110
for-in loop on strings	110
for-in loop on list	110
for-in loop on range	110
Iterable, iterator	111
for in loop with early end using break	111
for in loop skipping parts using continue	111
for in loop with break and continue	112
while loop	112
Infinite while loop	113
While with complex expression	113
While with break	114
While True	115
Duplicate input call	115
Eliminate duplicate input call	116
do while loop	116
while with many continue calls	117
Break out from multi-level loops	117
Exit vs return vs break and continue	117
Exercise: Print all the locations in a string	118
Exercise: Number guessing game	118
Exercise: Count unique characters	118
Exercise: Convert for-loop to while-loop	119
Solution: Print all the locations in a string	119
Solution 1 for Number Guessing	119
Solution 2 for Number Guessing (x)	120
Solution 3 for Number Guessing (s)	121
Solution for Number Guessing (debug)	122
Solution for Number Guessing (move)	122
Solution for Number Guessing (multi-game)	123
Solution: Count unique characters	125
Solution: Convert for-loop to while-loop	126
Formatted printing	127
format - sprintf	128
Examples using format - indexing	128

CONTENTS

Examples using format with names	129
Format columns	129
Examples using format - alignment	130
Format - string	130
Format characters and types	131
Format floating point number	131
f-strings (formatted string literals)	132
printf using old %-syntax	132
Format braces, bracket, and parentheses	133
Examples using format with attributes of objects	133
raw f-strings	133
Lists	135
Anything can be a list	135
Any layout	135
Lists	136
List slice with steps	137
Change a List	137
Change with steps	138
List assignment and list copy	138
Shallow vs. Deep copy of lists	139
join	139
join list of numbers	140
split	140
for loop on lists	141
in list	141
Where is the element in the list	142
Index improved	142
[	142
[	143
[	143
Remove element by index [	144
Remove first element of list	144
Remove several elements of list by index	145
Use list as a queue	145
Queue using deque from collections	145
Fixed size queue	146
List as a stack	147
stack with deque	147
Exercies: Queue	148
Exercise: Stack	149
Exercise: MasterMind	149
Solution: Queue with list	150

CONTENTS

Solution: Queue with deque	150
Solution: Reverse Polish calculator (stack) with lists	151
Solution: Reverse Polish calculator (stack) with deque	152
Solution: MasterMind	153
MasterMind to debug	154
Debugging Queue	156
sort	157
sort numbers	157
sort mixed	157
key sort	158
sort with sorted	158
sort vs. sorted	159
key sort with sorted	159
Sorting characters of a string	159
range	160
Looping over index	160
Enumerate lists	161
List operators	161
List of lists	162
List assignment	162
List documentation	163
tuple	163
Sort tuples	163
Exercise: color selector menu	164
Exercise: count digits	165
Exercise: Create list	165
Exercise: Count words	165
Exercise: Check if number is prime	166
Exercise: DNA sequencing	166
Solution: menu	166
Solution: count digits	167
Solution: Create list	168
Solution: Count words	169
Solution: Check if number is prime	170
Solution: DNA sequencing	170
Solution: DNA sequencing with filter	170
Solution: DNA sequencing with filter and lambda	171
[	171
append vs. extend	171
split and extend	172
Files	
File types: Text vs Binary	173

CONTENTS

Open vs. Read vs. Load	173
Binary files: Images	174
Reading an Excel file	174
Open and read file (easy but not recommended)	175
Open and read file using with (recommended)	175
Read file remove newlines	176
Filename on the command line	176
Filehandle with return	177
Read all the lines into a list	178
Read all the characters into a string (slurp)	178
Not existing file	178
Open file exception handling	179
Open many files - exception handling	180
Writing to file	180
Append to file	180
Binary mode	181
Does file exist? Is it a file?	181
Direct access of a line in a file	181
Exercise: count numbers	181
Exercise: strip newlines	181
Exercise: print lines with Report:	181
Exercise: color selector	181
Exercise: ROT13	181
Exercise: Combine lists	181
Solution: count numbers	181
Solution: strip newlines	181
Solution: print lines with Report:	181
Solution: color selector	181
Solution: Combine lists	181
Filehandle using with and not using it	181
Dictionary (hash)	191
What is a dictionary	191
When to use dictionaries	191
Dictionary	191
keys	191
Loop over keys	191
Loop over dictionary keys	191
Loop using items	192
values	192
Not existing key	193
Get key	193
Does the key exist?	194

CONTENTS

Does the value exist?	195
Delete key	195
List of dictionaries	196
Shared dictionary	196
immutable collection: tuple as dictionary key	197
immutable numbers: numbers as dictionary key	198
Sort dictionary by value	198
Sort dictionary keys by value	199
Insertion Order is kept	201
Change order of keys in dictionary - OrderedDict	201
Set order of keys in dictionary - OrderedDict	202
Exercise: count characters	202
Exercise: count words	203
Exercise: count words from a file	203
Exercise: Apache log	204
Exercise: Combine lists again	205
Exercise: counting DNA bases	205
Exercise: Count Amino Acids	205
Exercise: List of dictionaries	206
Exercise: Dictinoary of dictionaries	206
Exercise: Age limit with dictionaries	207
Solution: count characters	207
Default Dict	208
Solution: count characters with default dict	209
Solution: count words	210
Solution: count words in file	211
Solution: Apache log	211
Solution: Combine lists again	212
Solution: counting DNA bases	212
Solution: Count Amino Acids	213
Do not change dictionary in loop	214
Sets	216
sets	216
set operations	216
Creating a set	217
Creating an empty set	217
Adding an element to a set (add)	217
Merging one set into another set (update)	218
set intersection	218
set subset	219
set symmetric difference	220
set union	220

set relative complement

Functions (subroutines)

Why use functions?

Defining simple function

Passing positional parameters to a function

Function parameters can be named

Mixing positional and named parameters

Default values, optional parameters, optional parameters

Default value in first param

Several defaults, using names

Arbitrary number of arguments *

Fixed parameters before the others

Arbitrary key-value pairs in parameters **

Extra key-value pairs in parameters

Every parameter option

Duplicate declaration of functions (multiple signatures)

Pylint duplicate declaration

Return more than one value

Recursive factorial

Recursive Fibonacci

Non-recursive Fibonacci

Unbound recursion

Variable assignment and change - Immutable

Variable assignment and change - Mutable

Parameter passing of functions

Passing references

Function documentation

Sum ARGV

Copy-paste code

Copy-paste code fixed

Copy-paste code further improvement

Palindrome

Exercise: statistics

Exercise: recursive

Exercise: Tower of Hanoi

Exercise: Merge and Bubble sort

Exercise: Refactor previous solutions to use functions

Solution: statistics

Solution: recursive

Solution: Tower of Hanoi

Solution: Merge and Bubble sort

CONTENTS

Modules	246
Before modules	246
Create modules	246
path to load modules from - The module search path	247
sys.path - the module search path	247
Flat project directory structure	247
Absolute path	248
Relative path	248
Python modules are compiled	249
How "import" and "from" work?	249
Runtime loading of modules	249
Conditional loading of modules	250
Duplicate importing of functions	250
Script or library	251
Script or library - import	251
Script or library - from import	252
assert to verify values	252
mycalc as a self testing module	253
doctest	254
Scope of import	255
Export import	256
Export import with all	257
import module	258
Execute at import time	258
Import multiple times	258
Exercise: Number guessing	259
Exercises: Scripts and modules	259
Exercise: Module my_sum	259
Exercise: Convert your script to module	260
Exercise: Add doctests to your own code	260
Solution: Module my_sum	260
Regular Expressions	262
What are Regular Expressions (aka. Regexes)?	262
What are Regular Expressions good for?	262
Examples	262
Where can I use it ?	263
grep	263
Regexes first match	264
Match numbers	264
Capture	265
Capture more	265
Capture even more	266

CONTENTS

findall	264
findall with capture	265
findall with capture more than one	265
Any Character	269
Match dot	269
Character classes	270
Common character classes	270
Negated character class	270
Optional character	270
Regex 0 or more quantifier	270
Quantifiers	270
Quantifiers limit	270
Quantifiers on character classes	270
Greedy quantifiers	270
Minimal quantifiers	270
Anchors	270
Anchors on both end	270
Match ISBN numbers	270
Matching a section	270
Matching a section - minimal	270
Matching a section negated character class	270
DOTALL S (single line)	270
MULTILINE M	270
Two regex with logical or	28
Alternatives	28
Grouping and Alternatives	28
Internal variables	28
More internal variables	28
Regex DNA	28
Regex IGNORECASE	28
Regex VERBOSE X	28
Substitution	28
findall capture	28
Fixing dates	28
Duplicate numbers	28
Remove spaces	28
Replace string in assembly code	290
Full example of previous	290
Split with regex	291
Exercises: Regexes part 1	292
Exercise: Regexes part 2	292
Exercise: Sort SNMP numbers	293
Exercise: parse hours log file and give report	

CONTENTS

Exercise: Parse ini file	294
Exercise: Replace Python	295
Exercise: Extract phone numbers	295
Solution: Sort SNMP numbers	296
Solution: parse hours log file and give report	297
Solution: Processing INI file manually	299
Solution: Processing config file	300
Solution: Extract phone numbers	301
Regular Expressions Cheat sheet	301
Fix bad JSON	302
Fix very bad JSON	303
Raw string or escape	304
Remove spaces regex	305
Regex Unicode	305
Anchors Other example	306
PyCharm	308
PyCharm Intro	308
PyCharm configurations	308
PyCharm Project	308
PyCharm Files	308
PyCharm - run code	308
PyCharm Python console at the bottom left	308
Refactoring example with PyCharm	309
Python standard modules	310
Some Standard modules	310
sys	310
Writing to standard error (stderr)	311
Current directory (getcwd, pwd, chdir)	312
OS dir (mkdir, makedirs, remove, rmdir)	312
python which OS are we running on (os, platform)	312
Get process ID	313
OS path	313
Traverse directory tree - list directories recursively	313
os.path.join	314
Directory listing	314
expanduser - handle tilde ~	314
Listing specific files using glob	315
External command with system	315
subprocess	315
subprocess in the background	316
Accessing the system environment variables from Python	317

Set env and run command	2
shutil	2
time	2
sleep in Python	2
timer	2
Current date and time datetime now	2
Converting string to datetime	2
datetime arithmeticis	2
Rounding datetime object to nearest second	2
Signals and Python	2
Sending Signal	2
Catching Signal	2
Catching Ctrl-C on Unix	2
Catching Ctrl-C on Unix confirm	2
Alarm signal and timeouts	2
deep copy list	3
deep copy dictionary	3
Exercise: Catching Ctrl-C on Unix 2nd time	3
Exercise: Signals	3
Ctrl-z	3
JSON	33
JSON - JavaScript Object Notation	33
dumps	33
loads	33
dump	33
load	33
Round trip	33
Pretty print JSON	33
Sort keys in JSON	33
Set order of keys in JSON - OrderedDict	33
Exercise: Counter in JSON	33
Exercise: Phone book	33
Exercise: Processes	33
Solution: Counter in JSON	33
Solution: Phone book	34
Command line arguments with argparse	34
Modules to handle the command line	34
argparse	34
Basic usage of argparse	34
Positional argument	34
Many positional argument	34

CONTENTS

Convert to integers	343
Convert to integer	344
Named arguments	344
Boolean Flags	345
Short names	346
Exercise: Command line parameters	346
Exercise: argparse positional and named	346
argparse print help explicitly	346
Argparse xor - mutual exlucise, - only one - exactly one	347
Exception handling	349
Hierarchy of calls	349
Handling errors as return values	349
Handling errors as exceptions	350
A simple exception	350
Working on a list	351
Catch ZeroDivisionError exception	352
Module to open files and calculate something	353
File for exception handling example	353
Open files - exception	354
Handle divide by zero exception	355
Handle files - exception	355
Catch all the exceptions and show their type	356
List exception types	357
Exceptions	358
How to raise an exception	358
Stack trace	359
Exercies: Exception int conversion	360
Exercies: Raise Exception	361
Solution: Exception int conversion (specific)	361
Solution: Exception int conversion (all other)	362
Solution: Raise Exception	363
Classes - OOP - Object Oriented Programming	364
Why Object Oriented Programming?	364
Generic Object Oriented Programming terms	364
OOP in Python	364
OOP in Python (numbers, strings, lists)	364
OOP in Python (argparse)	365
Create a class	365
Import module containing class	366
Import class from module	366
Initialize a class - constructor, attributes	367

CONTENTS

Attributes are not special	367
Create Point class	368
Initialize a class - constructor, attributes	368
Methods	368
Stringify class	369
Inheritance	370
Inheritance - another level	371
Modes of method inheritance	372
Modes of method inheritance - implicit	372
Modes of method inheritance - override	372
Modes of method inheritance - extend	373
Modes of method inheritance - delegate - provide	374
Composition - Line	374
Some comments	375
Class in function	375
Serialization of instances with pickle	375
Quick Class definition and usage	376
Exercise: Add move_rad to based on radians	376
Exercise: Improve previous examples	377
Exercise: Polygon	377
Exercise: Number	378
Exercise: Library	378
Exercise: Bookexchange	378
Exercise: Represent turtle graphics	378
Solution - Polygon	378
PyPi - Python Package Index	380
What is PyPi?	380
Easy Install	380
pip	380
Upgrade pip	380
PYTHONPATH	380
Virtualenv	380
Virtualenv for Python 3	381
SQLite Database Access	382
SQLite	382
Connecting to SQLite database	382
Create TABLE in SQLite	382
INSERT data into SQLite database	383
SELECT data from SQLite database	384
A counter	384
MySQL	387

CONTENTS

Install MySQL support	387
Create database user (manually)	387
Create database (manually)	387
Create table (manually)	388
Connect to MySQL	388
Connect to MySQL and Handle exception	389
Select data	389
Select more data	390
Select all data fetchall	391
Select some data fetchmany	391
Select some data WHERE clause	392
Select into dictionaries	393
Insert data	394
Update data	394
Delete data	395
Exercise MySQL	396
Exercise: MySQL Connection	396
Solution: MySQL Connection	396
PostgreSQL	398
PostgreSQL install	398
Python and Postgresql	398
PostgreSQL connect	398
INSERT	398
INSERT (from command line)	399
SELECT	400
DELETE	400
SQLAlchemy	402
SQLAlchemy hierarchy	402
SQLAlchemy engine	402
SQLAlchemy autocommit	402
SQLAlchemy engine CREATE TABLE	402
SQLAlchemy engine INSERT	403
SQLAlchemy engine SELECT	403
SQLAlchemy engine SELECT all	404
SQLAlchemy engine SELECT fetchall	404
SQLAlchemy engine SELECT aggregate	405
SQLAlchemy engine SELECT IN	405
SQLAlchemy engine SELECT IN with placeholders	406
SQLAlchemy engine connection	406
SQLAlchemy engine transaction	406
SQLAlchemy engine using context managers	408

	408
Exercise: Create table	408
SQLAlchemy Metada	410
SQLAlchemy types	411
SQLAlchemy ORM - Object Relational Mapping	411
SQLAlchemy ORM create	412
SQLAlchemy ORM schema	413
SQLAlchemy ORM reflection	413
SQLAlchemy ORM INSERT after automap	414
SQLAlchemy ORM INSERT	414
SQLAlchemy ORM SELECT	415
SQLAlchemy ORM SELECT cross tables	415
SQLAlchemy ORM SELECT and INSERT	416
SQLAlchemy ORM UPDATE	417
SQLAlchemy ORM logging	417
Solution: Create table	418
Exercise: Inspector	419
SQLAlchemy CREATE and DROP	420
SQLAlchemy Notes	420
SQLAlchemy Meta SQLite CREATE	421
SQLAlchemy Meta Reflection	421
SQLAlchemy Meta INSERT	421
SQLAlchemy Meta SELECT	422
NoSQL	422
Types of NoSQL databases	423
MongoDB	423
MongoDB CRUD	423
Install MongoDB support	423
Python MongoDB insert	424
MongoDB CLI	425
Python MongoDB find	425
Python MongoDB find refine	426
Python MongoDB update	426
Python MongoDB remove (delete)	426
Python MongoDB replace	427
Python MongoDB upsert	428
Python Mongoddb: TypeError: upsert must be True or False	430
Redis	430
Redis CLI	430
Redis list keys	430
Redis set get	431
Redis incr	

CONTENTS

Redis incrby	431
Redis setex	431
Web client	433
urllib the web client	433
urllib2 the web client	433
httpbin.org	434
requests get	434
Download image using requests	434
Download image as a stream using requests	434
Download zip file	435
Extract zip file	435
Interactive Requests	435
requests get JSON	436
requests get JSON UserAgent	436
requests get JSON UserAgent	436
requests get header	437
requests change header	437
requests post	438
Tweet	438
API config file	439
bit.ly	439
Exercise: Combine web server and client	440
Python Web server	441
Hello world web	441
Dump web environment info	441
Web echo	442
Web form	443
Resources	444
Python Flask	445
Python Flask intro	445
Python Flask installation	445
Flask: Hello World	445
Flask: Run Hello World	446
Flask: testing hello world	447
Flask generated page - time	448
Flask generated page - time tested	449
Flask: Echo GET	450
Flask: Echo POST	451
Flask: templates	453
Flask: templates	453
Flask: templates with parameters	454

CONTENTS

Flask: runner	455
Exercise: Flask calculator	455
Static files	455
Flask Logging	456
Flask: Counter	457
Color selector without session	457
Session management	458
Flask custom 404 page	458
Flask Error page	459
Flask URL routing	460
Flask Path params	461
Flask Path params (int)	461
Flask Path params add (int)	462
Flask Path params add (path)	462
Jinja loop, conditional, include	463
Exercise: Flask persistent	464
Exercise: Flask persistent	464
Flask Exercises	465
Flask login	465
Flask JSON API	466
Flask and AJAX	467
Flask and AJAX	470
passlib	472
Flask Testing	472
Flask Deploy app	474
Flask Simple Authentication + test	475
Flask REST API	476
Flask REST API - Echo	476
Flask REST API - parameters in path	477
Flask REST API - parameter parsing	478
Flask REST API - parameter parsing - required	479
Networking	
Secure shell	481
ssh	481
ssh from Windows	481
Parallel ssh	481
telnet	482
prompt for password	482
Python nmap	483
ftp	483
Interactive shell	484
	486

The Python interactive shell	486
REPL - Read Evaluate Print Loop	486
Using Modules	487
Getting help	488
Exercise: Interactive shell	489
Testing Demo	490
How do you test your code?	490
What is testing?	490
What is testing really?	491
Testing demo tools	491
Testing demo methodology	491
Testing demo - AUT - Application Under Test	492
Testing demo - use the module	492
Testing demo: doctest	493
Testing demo: doctest with failure	494
Testing demo: Unittest success	496
Testing demo: Unittest failure	497
Testing demo: pytest using classes	498
Testing demo: pytest using classes - failure	499
Testing demo: pytest without classes	501
Testing demo: pytest run doctests	502
Testing demo: pytest run unittest	502
Exercise: Testing demo	502
Solution: Testing demo	503
Types in Python	504
mypy	504
Types of variables	504
Types of function parameters	504
Types used properly	505
TODO: mypy	505
Testing Intro	506
The software testing equasion	506
The software testing equasion (fixed)	506
The pieces of your software?	506
Manual testing	506
What to tests?	506
Continuous Integration	507
Functional programming	508
Functional programming	508
Iterators (Iterables)	508

range	50
range with list	51
range vs. list size	51
for loop with transformation	51
map	51
map delaying function call	51
map on many values	51
map with list	51
double with lambda	52
What is lambda in Python?	52
lambda returning tuple	52
map returning tuples	52
lambda with two parameters	52
map for more than one iterable	52
map on uneven lists	52
replace None (for Python 2)	52
map on uneven lists - fixed (for Python 2)	52
map mixed iterators	52
map fetch value from dict	52
Exercise: string to length	52
Exercise: row to length	52
Exercise: compare rows	52
Solution: string to length	52
Solution: row to length	52
Solution: compare rows	52
filter	52
filter with lambda	52
filter - map example	52
filter - map in one expression	52
Get indexes of values	52
reduce	53
reduce with default	53
zip	53
Creating dictionary from two lists using zip	53
all, any	53
Compare elements of list with scalar	53
List comprehension - double	53
List comprehension - simple expression	53
List generator	53
List comprehension	53
Dict comprehension	53
Lookup table with lambda	53
Read lines without newlines	53

CONTENTS

Read key-value pairs	538
Create index-to-value mapping in a dictionary based on a list of values	538
Exercise: min, max, factorial	539
Exercise: Prime numbers	539
Exercise: Many validator functions	539
Exercise: Calculator using lookup table	540
Exercise: parse file	540
Solution: min, max, factorial	541
Solution: Prime numbers	542
Solution: Many validator functions	542
Solution: Calculator using lookup table	542
map with condition	543
map with lambda	543
map with lambda with condition	544
List comprehension - complex	544
Iterators - with and without Itertools	545
Advantages of iterators and generators	545
The Fibonacci research institute	545
Fibonacci plain	545
Fibonacci copy-paste	546
Iterators Glossary	546
What are iterators and iterables?	548
A file-handle is an iterator	548
range is iterable but it is not an iterator	550
Iterator: a counter	550
Using iterator	551
Iterator without temporary variable	551
The type of the iterator	552
Using iterator with next	552
Mixing for and next	553
Iterable which is not an iterator	554
Iterator returning multiple values	554
Range-like iterator	555
Unbound or infinite iterator	556
Unbound iterator Fibonacci	558
Operations on Unbound iterator	558
itertools	559
itertools - count	559
itertools - cycle	560
Exercise: iterators - reimplement the range function	560
Exercise: iterators - cycle	560
Exercise: iterators - alter	560

Exercise: iterators - limit Fibonacci	561
Exercise: iterators - Fibonacci less memory	561
Exercise: read char	561
Exercise: read section	561
Exercise: collect packets	562
Exercise: compare files	563
Solution: iterators - limit Fibonacci	564
Solution: iterators - Fibonacci less memory	565
Solution: read section	565
Solution: compare files	566
Solution: collect packets	567
Generators and Generator Expressions	570
Generators Glossary	570
Iterators vs Generators	570
List comprehension and Generator Expression	570
List comprehension vs Generator Expression - less memory	571
List comprehension vs Generator Expression - lazy evaluation	573
Generator: function with yield - call next	574
Generators - call next	575
Generator with yield	575
Generators - fixed counter	576
Generators - counter	576
Generators - counter with parameter	576
Generators - my_range	577
Fibonacci - generator	578
Infinite series	579
Integers	579
Integers + 3	579
Integers + Integers	580
Filtered Fibonacci	580
The series.py	581
generator - unbound count (with yield)	582
iterator - cycle	583
Exercise: Alternator	583
Exercise: Prime number generator	584
Exercise: generator	584
Exercise: Tower of Hanoi	584
Exercise: Binary file reader	585
Exercise: File reader with records	585
Logging	585
Simple logging	587
	587

CONTENTS

Simple logging - set level	587
Simple logging to a file	588
Simple logging format	588
Simple logging change date format	589
getLogger	590
Time-based logrotation	590
Size-based logrotation	590
Closures	592
Counter local - not working	592
Counter with global	592
Create incrementors	593
Create internal function	593
Create function by a function	594
Create function with parameters	594
Counter closure	595
Make incrementor with def (closure)	596
Make incrementor with lambda	596
Exercise: closure bank	597
Solution: closure bank	597
Solution: counter with parameter	598
Decorators	600
Function assignment	600
Function inside other function	600
Decorator	601
Use cases for decorators in Python	602
A recursive Fibonacci	602
trace fibo	602
tron decorator	603
Decorate with direct call	603
Decorate with parameter	604
Decorator accepting parameter	604
Decorate function with any signature	605
Decorate function with any signature - implementation	605
Exercise: Logger decorator	606
Exercise: memoize decorator	606
Solution: Logger decorator	606
Solution: Logger decorator (testing)	607
Solution memoize decorator	608
Context managers (with statement)	610
Why use context managers?	610
Context Manager examples	611

CONTENTS

cd in a function	611
open in function	613
open in for loop	614
open in function using with	614
Plain context manager	615
Param context manager	615
Context manager that returns a value	616
Use my tempdir - return	617
Use my tempdir - exception	618
cwd context manager	618
tempdir context manager	619
Context manager with class	620
Context managers with class	621
Context manager: with for file	622
With - context managers	623
Exercise: Context manager	623
Exercise: Tempdir on Windows	624
Solution: Context manager	624
Advanced lists	
Change list while looping: endless list	626
Change list while looping	626
Copy list before iteration	626
for with flag	626
for else	627
enumerate	627
do while	628
list slice is copy	628
Advanced Exception handling	629
Exceptions else	630
Exceptions finally	630
Exit and finally	631
Catching exceptions	632
Home made exception	632
Home made exception with attributes	633
Home made exception hierarchy	634
Home made exception hierarchy - 1	635
Home made exception hierarchy - 2	635
Home made exception hierarchy - 3	636
Exercise: spaceflight with exceptions	636
Exercises: Raise My Exception	638
Solution: spaceflight with exceptions	639

CONTENTS

Solution: Raise My Exception	642
Exception finally return	643
Warnings	644
Warnings	644
CSV	645
Reading CSV the naive way	645
CSV with quotes and newlines	645
Reading a CSV file	646
CSV dialects	646
CSV to dictionary	647
Exercise: CSV	648
Solution: CSV	648
Excel	650
Spreadsheets	650
Python Excel	650
Create an Excel file from scratch	650
Worksheets in Excel	651
Add expressions to Excel	651
Format field	651
Number series and chart	652
Read Excel file	653
Update Excel file	653
Exercise: Excel	654
XML	655
XML Data	655
Expat - Callbacks	655
XML DOM - Document Object Model	656
XML SAX - Simple API for XML	657
SAX collect	658
XML elementtree	659
SciPy - for Scientific Computing in Python	661
Data Science tools in Python	661
Data Analysis resources	661
Python and Biology	663
Biopython	663
Biopython background	663
Bio python sequences	663
Download data	664

CONTENTS

	664
	665
Read FASTA, GenBank files	665
Search nucleotids	666
Download nucleotids	667
Exercise: Nucleotid	668
Biology background	668
Chemistry	668
Chemistry links	669
Bond length	669
Covalent radius	669
Python energy landscape explorer	669
Other chemistry links	670
numpy	670
What is NumPy	670
Numpy - vector	670
NumPy 2D arrays	671
Numpy - set type	671
NumPy arrays: ones and zeros	672
Numpy: eye	673
NumPy array random	673
NumPy Random integers	674
NumPy array type change by division (int to float)	674
Numpy: Array methods: transpose	675
Numpy: reference, not copy	676
Numpy: copy array	676
Numpy: Elementwise Operations on Arrays	677
Numpy: multiply, matmul, dot for vectors	677
Numpy: multiply, matmul, dot for vector and matrix	678
Numpy: multiply, matmul, dot for matrices	679
Numpy: casting - converting from strings to integer.	680
Numpy: indexing 1d array	680
Numpy: slice is a reference	681
Numpy: slice - copy	681
Numpy: abs value on a Numpy array	682
Numpy: Logical not on a Numpy array	683
Numpy: Vectorize a function	684
Numpy: Vectorize len	684
Numpy: Vectorize lambda	685
Numpy: Filtering array	686
Numpy: Filter matrix values	687
Numpy: Filter matrix rows	
Numpy: Stat	

CONTENTS

Numpy: Serialization	687
Numpy: Load from Matlab file	688
Numpy: Save as Matlab file	688
Numpy: Horizontal stack vectors (hstack)	688
Numpy: Append or vertically stack vectors and matrices (vstack)	688
Numpy uint8	689
Numpy int8	689
Pandas	691
Pandas	691
Planets	691
Pandas Planets - Dataframes	691
Pandas Stocks	692
Pandas Stocks	693
Merge Dataframes	693
Analyze Alerts	695
Analyze IFMetrics	695
Create Excel file for experiment with random data	695
Calculate Genome metrics	696
Calculate Genome metrics - add columns	697
Calculate Genome metrics - vectorized	698
Calculate Genome metrics - vectorized numpy	698
Genes using Jupyter	699
Combine columns	699
Pandas more	700
Pandas Series	701
Pandas Series with names	702
Matplotlib	704
About Matplotlib	704
Matplotlib Line	704
Matplotlib Line with dates	705
Matplotlib Simple Pie	706
Matplotlib Simple Pie with params	707
Matplotlib Pie	708
Matplotlib Pie 2	709
Plot, scatter, histogram	710
Seaborn	711
Seaborn use examples	711
Seaborn tip	711
Seaborn Anscombes Quartet	712
Jupyter notebooks	714

Jupyter on Windows	714
Jupyter on Linux and OSX	714
Jupyter add	714
Planets	715
Jupyter notebook Planets	715
Jupyter StackOverflow	716
Jupyter StackOverflow - selected columns	717
Jupyter processing chunks	717
Jupyter StackOverflow - selected rows	717
Jupyter StackOverflow - biggest countries (in terms of number of responses)	718
Jupyter StackOverflow - histogram	719
Jupyter StackOverflow - filter by country	719
Jupyter StackOverflow - OpenSourcer	719
Jupyter StackOverflow - cross tabulation	719
Jupyter StackOverflow - salaries	720
Jupyter StackOverflow - replace values	720
Jupyter StackOverflow - selected rows	720
Jupyter notebook Intellisense (TAB completion)	721
Jupyter examples	721
IPy Widgets	721
Testing	722
Traditional Organizations	722
Quality Assurance	722
Web age Organizations	722
TDD vs Testing as an Afterthought	722
Why test?	722
Testing Modes	722
Testing Applications	722
Testing What to test?	723
Testing in Python	723
Testing Environment	724
Testing Setup - Fixture	724
Testing Resources	724
Testing with unittest	725
Use a module	725
Test a module	725
The tested module	725
Testing - skeleton	726
Testing	726
Test examples	727
Testing with PyTest	728
	729

CONTENTS

Pytest features	729
Pytest setup	729
Testing with Pytest	729
Testing functions	730
Testing class and methods	730
Pytest - execute	731
Pytest - execute	731
Pytest simple module to be tested	731
Pytest simple tests - success	731
Pytest simple tests - success output	732
Pytest simple tests - failure	732
Pytest simple tests - failure output	732
Exercise: test math functions	733
Exercise: test this app	733
Exercise: test the csv module	734
Solution: Pytest test math functions	734
Solution: Pytest test this app	735
Solution: test the csv module	735
PyTest bank deposit	736
PyTest expected exceptions (bank deposit)	736
PyTest expected exceptions (bank deposit) - no exception happens	737
PyTest expected exceptions (bank deposit) - different exception is raised	738
PyTest expected exceptions	738
PyTest expected exceptions output	738
PyTest expected exceptions (text changed)	739
PyTest expected exceptions (text changed) output	739
PyTest expected exceptions (other exception)	739
PyTest expected exceptions (other exception) output	740
PyTest expected exceptions (no exception)	740
PyTest expected exceptions (no exception) output	741
PyTest: Multiple Failures	741
PyTest: Multiple Failures output	742
PyTest Selective running of test functions	742
PyTest: stop on first failure	742
Pytest: expect a test to fail (xfail or TODO tests)	742
Pytest: expect a test to fail (xfail or TODO tests)	743
PyTest: show xfailed tests with -rx	743
Pytest: skipping tests	744
Pytest: show skipped tests with -rs	745
Pytest: show extra test summary info with -r	745
Pytest: skipping tests output in verbose mode	745
Pytest verbose mode	746
Pytest quiet mode	746

	747
PyTest print STDOUT and STDERR using -s	747
PyTest failure reports	747
PyTest compare numbers	747
PyTest compare numbers relatively	748
PyTest compare strings	748
PyTest compare long strings	749
PyTest is one string in another strings	750
PyTest test any expression	750
PyTest element in list	750
PyTest compare lists	751
PyTest compare short lists	751
PyTest compare short lists - verbose output	752
PyTest compare dictionaries	752
PyTest compare dictionaries output	753
PyTest Fixtures	753
PyTest Fixture setup and teardown	754
PyTest Fixture setup and teardown output	755
PyTest: Class setup and teardown	756
PyTest: Class setup and teardown output	756
Pytest Dependency injection	757
Pytest fixture - tmpdir	757
Pytest capture STDOUT and STDERR with capsys	757
Pytest Fixture - home made fixtures	758
More fixtures	759
Pytest: Mocking - why?	761
Pytest: Mocking - what?	761
Pytest: One dimensional spaceflight	761
Pytest: Mocking input and output	763
Pytest: Mocking random	763
Pytest: Flask echo	764
Pytest: testing Flask echo	765
PyTest: Run tests in parallel with xdist	766
PyTest: Order of tests	766
PyTest: Randomize Order of tests	766
PyTest: Force default order	766
PyTest: no random order	767
Anagram on the command line	767
PyTest testing CLI	767
PyTest test discovery	767
PyTest test discovery - ignore some tests	768
PyTest select tests by name	769
PyTest select tests by marker	770
PyTest: Test Coverage	771
	772

Exercise: module	772
Exercise: Open Source	773
Pytest resources	773
Pytest and tempdir	773
PyTest compare short lists - output	774
PyTest with parameter	775
PyTest with parameters	776
Pytest reporting in JUnit XML format	776
No test selected	777
Advanced functions	778
Variable scopes	778
Name resolution order (LEGB)	778
Scoping: global seen from function	778
Assignment creates local scope	778
Local scope gone wrong	779
Changing global variable from a function	780
Global variables mutable in functions	780
Scoping issues	780
sub in sub	781
Scoping sub in sub (enclosing scope)	781
Function objects	782
Functions are created at run time	782
Mutable default	783
Use None as default parameter	783
Inner function created every time the outer function runs	784
Static variable	785
Static variable in generated function	785
Inspect	786
Variable number of function arguments	788
Python function arguments - a reminder	788
Functions with unknown number of arguments	788
Variable length argument list with * and **	789
Passing arguments as they were received (but incorrectly)	789
Unpacking args before passing them on	790
Exercise: implement the my_sum function	790
Solution: implement the my_sum function	791
Exercise: implement the reduce function	791
Solution: implement the reduce function	791
Exercise: sort pairs	792
Solution: sort pairs	792
Python Packages	794

Why Create package	794
Create package	794
Internal usage	795
use module in package - relative path	795
use package (does not work)	796
package importing (and exporting) module	796
use package (module) with import	796
use package with import	797
Creating an installable Python package	797
Create tar.gz file	798
Install Package	799
Dependencies	800
Add README file	800
Add README file (setup.py)	801
Include executables	802
Add tests	802
Add tests calc	803
Add tests all	803
setup.py	803
Run tests and create package	804
Packaging applications (creating executable binaries)	804
Using PyInstaller	804
Other PyInstaller examples	804
Other	805
Py2app for Mac	805
Exercise: package	806
Exercise: create executable	806
Ctypes	807
ctypes - hello	808
concat	809
links	809
Advanced OOP	809
Class count instances	810
Class Attributes	811
Class Attributes in Instances	811
Attributes with method access	812
Instance Attribute	813
Methods are class attributes	814
Monkey patching	815
Classes: instance method	
Class methods and class attributes	

CONTENTS

Classes: constructor	816
Class methods - alternative constructor	817
Abstract Base Class	818
Abstract Base Class with abc	819
ABC working example	819
ABC - cannot instantiate the base-class	820
ABC - must implement methods	820
Use Python @property to fix bad interface (the bad interface)	821
Use Python @property to fix bad interface (first attempt)	821
Use Python @property to fix bad API	822
Use Python @property decorator to fix bad API	823
Use Python @property for value validation	824
class and static methods	825
Destructor: del	827
Destructor delayed	828
Destructor delayed for both	828
Operator overloading	829
Operator overloading methods	830
Exercise: rectangular	830
Exercise: SNMP numbers	831
Exercise: Implement a Gene inheritance model combining DNA	831
Exercise: imaginary numbers - complex numbers	831
Solution: Rectangular	832
Solution: Implement a Gene inheritance model combining DNA	835
Instance counter	836
2to3	837
Convertig from Python 2 to Python 3	837
division	837
print in Python 2	837
print in Python 3	837
input and raw_input	838
Code that works on both 2 and 3	838
Compare different types	838
Octal numbers	839
2to3 Resources	839
Design Patterns	840
What are Design Patterns?	840
Don't replace built-in objects	840
Facade - simple interface to complex system	840
Monkey Patching	841
Creation DPs "Just One"	842

CONTENTS

Singleton	842
Monostate (Borg)	843
Dispatch table	844
Parallel	844
Types of Problems	844
Types of solutions	844
How many parallels to use?	844
Dividing jobs	845
Performance Monitoring	846
Threads	846
Python Threading docs	846
Threaded counters	847
Simple threaded counters	849
Simple threaded counters (parameterized)	850
Pass parameters to threads - Counter with attributes	852
Create a central counter	852
Lock - acquire - release	853
Counter - plain	853
GIL - Global Interpreter Lock	853
Thread load	853
Exercise: thread files	853
Exercise: thread URL requests.	853
Exercise: thread queue	853
Solution: thread queue	86
Solution: thread URL requests.	86
Forking	86
Fork	86
Forking	86
Fork skeleton	86
Fork with load	86
Fork load results	86
Marshalling / Serialization	86
Fork with random	86
Exercise: fork return data	86
Solution: fork return data	86
Asynchronous programming with AsyncIO	87
Sync chores	87
Async chores	87
Explanation	87
Coroutines	87

CONTENTS

More about asyncio	876
Async files	876
Asynchronous programming with Twisted	877
About Twisted	877
Echo	877
Echo with log	878
Simple web client	879
Web client	880
Multiprocess	883
Multiprocess CPU count	883
Multiprocess Process	883
Multiprocess N files: Pool	883
Multiprocess load	884
Multiprocess: Pool	885
Multiprocess load async	886
Multiprocess and logging	887
Exercise: Process N files in parallel	888
Exercise: Process N Excel files in parallel	888
Exercise: Fetch URLs in parallel	889
Exercise: Fetch URLs from one site	891
Solution: Fetch URLs in parallel	892
Multitasking	895
What is Multitasking?	895
Multitasking example	895
Multitasking example with wait	896
Multitasking - second loop waits for first one	897
Multitasking counter	898
Multitasking counter with thread locking	899
Improving Performance - Optimizing code	900
Problems	900
Optimization strategy	900
Locate the source of the problem	900
Optimizing tactics	900
DSU: Decorate Sort Undecorate	901
Profile code	901
Slow example	901
profile slow code	903
cProfile slow code	903
Benchmarking	904
Benchmarking subs	905

Levenshtein distance	905
Generate words	906
Levenshtein - pylev	907
Levenshtein - editdistance	907
Editdistance benchmark	907
A Tool to Generate text files	909
Count characters	910
Memory leak	911
Garbage collection	912
Weak reference	913
Exercise: benchmark list-comprehension, map, for	913
Exercise: Benchmark Levenshtein	913
Exercise: sort files	913
Exercise: compare split words	913
Exercise: count words	914
GUI with Python/Tk	916
Sample Tk app	916
GUI Toolkits	920
Installation	920
Python Tk Documentation	921
Python Tk Button	921
Python Tk Button with action	922
Python Tk Label	922
Python Tk Label - font size and color	923
Python Tk Keybinding	923
Python Tk Entry (one-line text entry)	924
Python Tk Entry for passwords and other secrets (hidden text)	925
Python Tk Checkbox	925
Python Tk Radiobutton	926
Python Tk Listbox	927
Python Tk Listbox Multiple	927
Python Tk Menubar	928
Python Tk Text	930
Python Tk Dialogs	930
Python Tk Filedialog	930
Python Tk messagebox	932
Python Tk Combobox	933
Python Tk OptionMenu	934
Python Tk Scale	935
Python Tk Progressbar	936
Python Tk Frame	936
Not so Simple Tk app with class	938

CONTENTS

Tk: Hello World	938
Tk: Quit button	939
Tk: File selector	940
Tk: Checkbox	941
Tk: Runner	942
Tk: Runner with threads	944
Getting started with Tk	948
Exercise: Tk - Calculator one line	949
Exercise: Tk Shopping list	949
Exercise: Tk TODO list	950
Exercise: Tk Notepad	950
Exercise: Tk Copy files	950
Exercise: Tk Implement Master Mind board	950
Exercise: Tk	950
Solution: Tk - Calculator one line	951
Solution: Tk Implement Master Mind board	953
Solution: Tk	953
Solution: Tk Notepad	954
Simple file dialog	956
Python Pitfalls	957
Reuse of existing module name	957
Use the same name more than once	957
Compare string and number	958
Compare different types	959
Sort mixed data	959
Linters	961
Static Code Analyzis - Linters	961
PEP8	961
F811 - redefinition of unused	961
Warn when Redefining functions	961
Python .NET	963
IronPython	963
Use .NET libraries from Python	963
Python and .NET console	964
Python and .NET examples	964
Exercise Python and .NET	966
Python and Java	967
Jython	967
Calling Java from Python	967

Jython - Python running on the JVM

Jython Installation	91
Jython Installation	91
Jython load Java class	91
Jython load Java class in code	91
Jython test Java class	91

PIL - Pillow

Install Pillow	91
Create First Image	91
Write Text on Image	91
Select font for Text on Image	91
Font directories	91
Get size of an Image	91
Get size of text	91
Resize an existing Image	91
Crop an existing Image	91
Combine two images	91
Rotated text	91
Rotated text in top-right corner	91
Embed image (put one image on another one)	91
Draw a triangle	91
Draw a triangle and write text in it	91
Draw a triangle and write rotated text in it	91
Draw a rectangular	91
Draw a rectangle	91
Draw circle	91
Draw heart	91
Rectangle with rounded corners	91
TODO	91

FAQ

How not to name example scripts?	91
Platform independent code	91
How to profile a python code to find causes of slowness?	91
pdb - Python Debugger	98
Avoid Redefining functions	98

Appendix

print_function	981
Dividers (no break or continue)	981
Lambdas	982
Abstract Class	982
Remove file	983

CONTENTS

Modules: more	984
import hooks	985
Python resources	985
Progress bar	985
from future	985
Variable scope	986
scope	987
type	989
Look deeper in a list	990
Exercise: iterators - count	990
Simple function (before generators)	990
Other slides	992
Other slides	992
Atom for Python	993
IDLE - Integrated DeveLopment Environment	993
sh-bang - executable on Linux/Apple	993
Strings as Comments	994
pydoc	994
How can I check if a string can be converted to a number?	995
Spyder Intro	995
Interactive Debugging	996
Parameter passing	996
Command line arguments and main	996
Infinite loop	996
break	997
continue	997
While with many conditions	998
while loop with many conditions	998
Format with conversion (stringification with str or repr)	999
Name of the current function in Python	1000
Name of the caller function in Python	1000
Stack trace in Python using inspect	1002
Module Fibonacci	1002
PyTest - assertion	1003
PyTest - failure	1003
PyTest - list	1004
SAX with coroutine	1006
Getting the class name of an object	1007
Inheritance - super	1008
Inheritance - super - other class	1009
iterator - pairwise	1009
iterator - grouped	1009

CONTENTS

itertools - groupby	.1010
Circular references	.1010
Context managers: with (file) experiments	.1011
itertools - izip	.1011
mixing iterators	.1012
mixing iterators	.1012
itertools - pairwise	.1013
itertools - grouped	.1013
range vs xrange in Python	.1014
profile (with hotshot) slow code	.1014
Abstract Base Class without abc	.1015
Abstract Base Class with abc Python 2 ?	.1016
Abstract Base Class with metaclass	.1017
Create class with metaclass	.1019
Python Descriptors	.1022
alter iterator	.1022
Create a counter queue	.1022
A Queue of tasks	.1023
Filtered Fibonacci with ifilter	.1024
Python from .NET	.1024