

Table of Contents

Preface.....	xi
--------------	----

Part I. The Rosalind.info Challenges

1. Tetranucleotide Frequency: Counting Things.....	3
Getting Started	4
Creating the Program Using new.py	5
Using argparse	7
Tools for Finding Errors in the Code	10
Introducing Named Tuples	12
Adding Types to Named Tuples	15
Representing the Arguments with a NamedTuple	16
Reading Input from the Command Line or a File	18
Testing Your Program	20
Running the Program to Test the Output	23
Solution 1: Iterating and Counting the Characters in a String	25
Counting the Nucleotides	26
Writing and Verifying a Solution	28
Additional Solutions	30
Solution 2: Creating a count() Function and Adding a Unit Test	30
Solution 3: Using str.count()	34
Solution 4: Using a Dictionary to Count All the Characters	35
Solution 5: Counting Only the Desired Bases	38
Solution 6: Using collections.defaultdict()	39
Solution 7: Using collections.Counter()	41
Going Further	42
Review	42

2. Transcribing DNA into mRNA: Mutating Strings, Reading and Writing Files.	45
Getting Started	46
Defining the Program's Parameters	47
Defining an Optional Parameter	47
Defining One or More Required Positional Parameters	48
Using nargs to Define the Number of Arguments	49
Using argparse.FileType() to Validate File Arguments	49
Defining the Args Class	50
Outlining the Program Using Pseudocode	51
Iterating the Input Files	52
Creating the Output Filenames	52
Opening the Output Files	54
Writing the Output Sequences	55
Printing the Status Report	57
Using the Test Suite	57
Solutions	60
Solution 1: Using str.replace()	60
Solution 2: Using re.sub()	62
Benchmarking	64
Going Further	65
Review	65
3. Reverse Complement of DNA: String Manipulation.	67
Getting Started	68
Iterating Over a Reversed String	70
Creating a Decision Tree	72
Refactoring	73
Solutions	74
Solution 1: Using a for Loop and Decision Tree	75
Solution 2: Using a Dictionary Lookup	75
Solution 3: Using a List Comprehension	78
Solution 4: Using str.translate()	78
Solution 5: Using Bio.Seq	81
Review	82
4. Creating the Fibonacci Sequence: Writing, Testing, and Benchmarking Algorithms.	83
Getting Started	84
An Imperative Approach	89
Solutions	91
Solution 1: An Imperative Solution Using a List as a Stack	91
Solution 2: Creating a Generator Function	93
Solution 3: Using Recursion and Memoization	96

Benchmarking the Solutions	100
Testing the Good, the Bad, and the Ugly	102
Running the Test Suite on All the Solutions	103
Going Further	109
Review	109
5. Computing GC Content: Parsing FASTA and Analyzing Sequences.....	111
Getting Started	112
Get Parsing FASTA Using Biopython	115
Iterating the Sequences Using a for Loop	118
Solutions	120
Solution 1: Using a List	120
Solution 2: Type Annotations and Unit Tests	123
Solution 3: Keeping a Running Max Variable	127
Solution 4: Using a List Comprehension with a Guard	129
Solution 5: Using the filter() Function	130
Solution 6: Using the map() Function and Summing Booleans	130
Solution 7: Using Regular Expressions to Find Patterns	131
Solution 8: A More Complex find_gc() Function	132
Benchmarking	134
Going Further	134
Review	135
6. Finding the Hamming Distance: Counting Point Mutations.....	137
Getting Started	138
Iterating the Characters of Two Strings	141
Solutions	142
Solution 1: Iterating and Counting	142
Solution 2: Creating a Unit Test	143
Solution 3: Using the zip() Function	145
Solution 4: Using the zip_longest() Function	147
Solution 5: Using a List Comprehension	148
Solution 6: Using the filter() Function	149
Solution 7: Using the map() Function with zip_longest()	150
Solution 8: Using the starmap() and operator.ne() Functions	151
Going Further	153
Review	153
7. Translating mRNA into Protein: More Functional Programming.....	155
Getting Started	155
K-mers and Codons	157
Translating Codons	160

Solutions	161
Solution 1: Using a for Loop	161
Solution 2: Adding Unit Tests	162
Solution 3: Another Function and a List Comprehension	165
Solution 4: Functional Programming with the map(), partial(), and takewhile() Functions	167
Solution 5: Using Bio.Seq.translate()	169
Benchmarking	170
Going Further	170
Review	170
8. Find a Motif in DNA: Exploring Sequence Similarity.....	171
Getting Started	171
Finding Subsequences	173
Solutions	175
Solution 1: Using the str.find() Method	176
Solution 2: Using the str.index() Method	177
Solution 3: A Purely Functional Approach	179
Solution 4: Using K-mers	181
Solution 5: Finding Overlapping Patterns Using Regular Expressions	183
Benchmarking	184
Going Further	185
Review	185
9. Overlap Graphs: Sequence Assembly Using Shared K-mers.....	187
Getting Started	188
Managing Runtime Messages with STDOUT, STDERR, and Logging	192
Finding Overlaps	195
Grouping Sequences by the Overlap	196
Solutions	200
Solution 1: Using Set Intersections to Find Overlaps	200
Solution 2: Using a Graph to Find All Paths	203
Going Further	208
Review	208
10. Finding the Longest Shared Subsequence: Finding K-mers, Writing Functions, and Using Binary Search.....	211
Getting Started	211
Finding the Shortest Sequence in a FASTA File	213
Extracting K-mers from a Sequence	215
Solutions	217
Solution 1: Counting Frequencies of K-mers	217

Solution 2: Speeding Things Up with a Binary Search	220
Going Further	226
Review	226
11. Finding a Protein Motif: Fetching Data and Using Regular Expressions.	227
Getting Started	227
Downloading Sequences Files on the Command Line	230
Downloading Sequences Files with Python	233
Writing a Regular Expression to Find the Motif	235
Solutions	237
Solution 1: Using a Regular Expression	237
Solution 2: Writing a Manual Solution	239
Going Further	244
Review	244
12. Inferring mRNA from Protein: Products and Reductions of Lists.	245
Getting Started	245
Creating the Product of Lists	247
Avoiding Overflow with Modular Multiplication	249
Solutions	251
Solution 1: Using a Dictionary for the RNA Codon Table	251
Solution 2: Turn the Beat Around	257
Solution 3: Encoding the Minimal Information	259
Going Further	260
Review	261
13. Location Restriction Sites: Using, Testing, and Sharing Code.	263
Getting Started	264
Finding All Subsequences Using K-mers	266
Finding All Reverse Complements	267
Putting It All Together	267
Solutions	268
Solution 1: Using the zip() and enumerate() Functions	268
Solution 2: Using the operator.eq() Function	270
Solution 3: Writing a revp() Function	271
Testing the Program	272
Going Further	274
Review	274
14. Finding Open Reading Frames.	275
Getting Started	275
Translating Proteins Inside Each Frame	277

Finding the ORFs in a Protein Sequence	279
Solutions	280
Solution 1: Using the str.index() Function	280
Solution 2: Using the str.partition() Function	282
Solution 3: Using a Regular Expression	284
Going Further	286
Review	286

Part II. Other Programs

15. Seqmagique: Creating and Formatting Reports.....	289
Using Seqmagick to Analyze Sequence Files	290
Checking Files Using MD5 Hashes	291
Getting Started	293
Formatting Text Tables Using tabulate()	295
Solutions	296
Solution 1: Formatting with tabulate()	296
Solution 2: Formatting with rich	303
Going Further	305
Review	306
16. FASTX grep: Creating a Utility Program to Select Sequences.....	307
Finding Lines in a File Using grep	308
The Structure of a FASTQ Record	308
Getting Started	311
Guessing the File Format	315
Solution	317
Going Further	327
Review	327
17. DNA Synthesizer: Creating Synthetic Data with Markov Chains.....	329
Understanding Markov Chains	329
Getting Started	332
Understanding Random Seeds	335
Reading the Training Files	337
Generating the Sequences	340
Structuring the Program	343
Solution	343
Going Further	347
Review	347

18. FASTX Sampler: Randomly Subsampling Sequence Files.....	349
Getting Started	349
Reviewing the Program Parameters	350
Defining the Parameters	352
Nondeterministic Sampling	354
Structuring the Program	356
Solutions	356
Solution 1: Reading Regular Files	357
Solution 2: Reading a Large Number of Compressed Files	358
Going Further	360
Review	360
19. Blastomatic: Parsing Delimited Text Files.....	361
Introduction to BLAST	361
Using csvkit and csvchk	364
Getting Started	368
Defining the Arguments	371
Parsing Delimited Text Files Using the csv Module	373
Parsing Delimited Text Files Using the pandas Module	377
Solutions	383
Solution 1: Manually Joining the Tables Using Dictionaries	383
Solution 2: Writing the Output File with csv.DictWriter()	384
Solution 3: Reading and Writing Files Using pandas	385
Solution 4: Joining Files Using pandas	387
Going Further	390
Review	390
A. Documenting Commands and Creating Workflows with make.....	391
B. Understanding \$PATH and Installing Command-Line Programs.....	405
Epilogue.....	409
Index.....	411