Introduction to Python

BCH 519
Spring 2015

Andrew E. Bruno
aebruno2@buffalo.edu

Chapters 3–5

Exercises and Solutions

Chapter 3: Reading and writing files

Exercises and Solutions

Splitting genomic DNA

- Here's a short section of genomic DNA:

```
ATCGATCGATCGATCGACTGACTAGTCATAGCTATGCATGTAGCTACTCGATCGATCGATCGATCGATCGATCGATCGA
TCGATCATGCTATCATCGATCGATATCGATGCATCGACTACTAT
```

  It comprises two exons and an intron. The first exon runs from the start of the sequence to the sixty-third character, and the second exon runs from the ninety-first character to the end of the sequence. Write a program that will split the genomic DNA into coding and non-coding parts, and write these sequences to two separate files.

Solution 1

```
seq = 'ATCGATCGATCGATCGACTGACTAGTCATAGCTATGCATGTAGCTACTCGATCGATCGATCGA'
seq += 'TCGATCGATCGATCGATCGATCATGCTATCATCGATCGATATCGATGCATCGACTACTAT'

exon1 = seq[0:62]
exon2 = seq[90:]
intron = seq[63:90]

with open('coding.txt', 'w') as coding:
    coding.write(exon1.upper() + exon2.upper() + '\n')

with open('non-coding.txt', 'w') as non_coding:
    non_coding.write(intron.lower() + '\n')
```

Chapter 4: Lists and loops

Exercises and Solutions

Processing DNA in a file

- The file /data/dna/input.txt contains a number of DNA sequences, one per line. Each sequence starts with the same 14 base pair fragment - a sequencing adapter that should have been removed. Write a program that will (a) trim this adapter and write the cleaned sequences to a new file and (b) print the length of each sequence to the screen.

Solution 1

```
with open('input.txt') as fin, open('output.txt', 'w') as fout:
    for line in fin:
        # Strip out adapter
        dna = line[14:]

        # Print seq to file
        fout.write(dna)

        # Print len to screen
        print len(dna)
```

Multiple exons from genomic DNA

- The file /data/dna/genomic_dna.txt contains a section of genomic DNA, and the file /data/dna/exons.txt contains a list of start/stop positions of exons. Each exon is on a separate line and the start and stop positions are separated by a comma. Write a program that will extract the exon segments, concatenate them, and write them to a new file. *Note: You can assume start/stop are 0 based*

Solution 1

```
dna = ''
with open('genomic_dna.txt') as fin:
    for line in fin:
        dna += line.strip()

exons = ''
with open('exons.txt') as fin:
    for line in fin:
        start,stop = line.strip().split(',')

        exons += dna[int(start):int(stop)]

with open('output.txt', 'w') as fout:
    fout.write(exons + '\n')
```

Chapter 5: Writing our own functions

Exercises and Solutions

Percentage of amino acid residues, part 1

- Write a function that takes two arguments - a protein sequence and an amino acid residue code - and returns the percentage of the protein that the amino acid makes up. Use the following assertions to test your function:

```
assert my_function("MSRSLLLRFLLFLLLLPPLP", "M") == 5
assert my_function("MSRSLLLRFLLFLLLLPPLP", "r") == 10
assert my_function("MSRSLLLRFLLFLLLLPPLP", "L") == 50
assert my_function("MSRSLLLRFLLFLLLLPPLP", "Y") == 0
```

Solution 1

```
from __future__ import division

def amino_acid_pct(seq, residue):
    return round((seq.upper().count(residue.upper()) / len(seq) ) * 100)

assert amino_acid_pct("MSRSLLLRFLLFLLLLPPLP", "M") == 5
assert amino_acid_pct("MSRSLLLRFLLFLLLLPPLP", "r") == 10
assert amino_acid_pct("MSRSLLLRFLLFLLLLPPLP", "L") == 50
assert amino_acid_pct("MSRSLLLRFLLFLLLLPPLP", "Y") == 0
```

Percentage of amino acid residues, part 2

- Modify the function from part one so that it accepts a list of amino acid residues rather than a single one. If no list is given, the function should return the percentage of hydrophobic amino acid residues (A, I, L, M, F, W, Y and V). Your function should pass the following assertions:

```
assert my_function("MSRSLLLRFLLFLLLLPPLP", ["M"]) == 5
assert my_function("MSRSLLLRFLLFLLLLPPLP", ['M', 'L']) == 55
assert my_function("MSRSLLLRFLLFLLLLPPLP", ['F', 'S', 'L']) == 70
assert my_function("MSRSLLLRFLLFLLLLPPLP") == 65
```

Solution 1

```
from __future__ import division

def amino_acid_pct(seq, residue = None):
    if residue is None:
        residue = ['A', 'I', 'L', 'M', 'F', 'W', 'Y', 'V']

    count = 0
    for r in residue:
        count += seq.upper().count(r.upper())

    return round((count / len(seq)) * 100)

assert amino_acid_pct("MSRSLLLRFLLFLLLLPPLP", ["M"]) == 5
assert amino_acid_pct("MSRSLLLRFLLFLLLLPPLP", ['M', 'L']) == 55
assert amino_acid_pct("MSRSLLLRFLLFLLLLPPLP", ['F', 'S', 'L']) == 70
assert amino_acid_pct("MSRSLLLRFLLFLLLLPPLP") == 65
```

Homework #3

- Due: 2015-02-19 09:00:00