MCS 260 – Introduction to Computer Science – Fall 2020 – Emily Dumas

Week 4 Worksheet Solutions

Note. Most coding problems admit many correct answers. This document shows an example of a correct solution to each problem.

Problems.

(1) Suppose that `L` is a list whose elements are sequences. Generate a new list `M` where `M[i]` is equal to the length of `L[i]` (an integer). Thus if `L = [ [5,6], "Fuji" ]` then the result should be `M = [2,4]`; but your code should work for any `L` whose elements are sequences, not just in this one example.

(a) Use a for loop to do this.

Answer:

```
M = []
for seq in L:
    M.append(len(seq))
```

(b) Use a list comprehension.

Answer:

```
M = [len(seq) for seq in L]
```

(2) Write a function `with_spaces()` that takes one parameter, an iterable of strings, and returns a list of the strings in this iterable that contain a space character. So, for example, if we set

`L = [ "banana", "apple", "green pear", "guava", "red dragonfruit" ]`

then `with_spaces(L)` should evaluate to

`[ "green pear", red dragonfruit" ]`

Answer:

```
def with_spaces(L):
    """Take in a list of strings, and return a list of the strings
    that contain spaces.
    """
    R = []
    for word in L:
        if " " in word:
            R.append(word)
    return R
```

(3) In each part of this problem, write code that uses the following list of tuples:

```
coursedata = [ ("MCS",260,"Intro. to comp. sci."),
               ("MCS",275,"Prog. tools and file mgmt."),
               ("MATH",180,"Calculus I"),
               ("MATH",320,"Linear Algebra I"),
               ("MATH",549,"Differentiable Manifolds I"), ]
```

Thus you probably want to copy and paste this into a source file, or download it from

https://dumas.io/teaching/2020/fall/mcs260/samplecode/coursedata.py

(a) Write a for loop that iterates over this list and prints all of the course numbers, i.e.

```
MCS 260
MCS 275
MATH 180
MATH 320
MATH 549
```

Answer:

```
for course in coursedata:
    print(course[0], course[1])
```

(b) Write a list comprehension that iterates over `coursedata` and yields

```
[ "MCS 260", "MCS 275", "MATH 180", "MATH 320", "MATH 549" ]
```

Answer:

```
B = [course[0]+" "+str(course[1]) for course in coursedata]
```

(c) Write a for loop that prints the data for each MCS course in the following format:

```
Course Number: MCS 260
Description: Intro. to comp. sci.
```

Answer:

```
for course in coursedata:
    if course[0] == "MCS":
        print("Course Number:", course[0], course[1])
        print("Description:", course[2])
```

(d) Write a list comprehension that is analogous to part (c), but yields a list of strings, one for each MCS course. For example the first string would be

```
"Course Number: MCS 260\nDescription: Intro. to comp. sci.\n"
```

Answer:

```
# Note: A list comprehension can be split between multiple lines,
# and we do that here because it would be very long if on one line
D = [ "Course Number: " + c[0] + " " + str(c[1])
      + "\nDescription: " + c[2] + "\n"
      for c in coursedata if c[0]=="MCS" ]
```

(4) Using the list of lists below, write a for loop inside of a for loop that will print the even numbers that occur as elements of elements of `L`.

```
L = [ [3,1,2], [9,9,6], [3,0,4,1] ]
```

That is, the output should be:

```
2
6
0
4
```

Answer:

```
for v in L:
    for n in v:
        if n%2 == 0:
            print(n)
```

(5) Write a function `opening(...)` to generate the first line of a letter or memo. It should take parameters `fullname`, `salutation`, and `greeting` (in that order). The parameter `salutation` should have default value `""`, and the parameter `greeting` should have default value `"Dear"`. This function should print the greeting, followed by a space, an optional salutation, the fullname, and a comma. Examples:

- `opening("Grace Hopper")` prints
  `Dear Grace Hopper,`
- `opening("Emily Dumas",greeting="Howdy")` prints
  `Howdy Emily Dumas,`
- `opening("Marie Curie",salutation="Dr.",greeting="To the esteemed")` prints
  `To the esteemed Dr. Marie Curie,`

Hint: How do you avoid printing two spaces between the greeting and the fullname if the salutation is the empty string?

Answer:

```
def opening(fullname, salutation="", greeting="Dear"):
    """Generates the first line of a letter or memo"""
    if salutation == "":
        print(greeting, fullname + ",")
    else:
        print(greeting, salutation, fullname + ",")
```

(6) Write a function that takes a list of floats and returns their average (mean). (As a reminder, the mean of real numbers $x_1, x_2, \ldots, x_n$ is defined as $\frac{1}{n}(x_1 + x_2 + \cdots + x_n)$.)

Answer:

```
def average(L):
    """Returns the average value of a list of real numbers"""
    sum = 0
    for n in L:
        sum = sum + n
    return sum / len(L)
```

(7) Suppose that `cmds` is an iterable of strings. Write a loop that will print the elements of `cmds` in order, stopping with the first one that is equal to `"stop"`, `"exit"`, or `"end"`.

Answer:

```
stopwords = ["stop", "exit", "end"]
for cmd in cmds:
    print(cmd)
```

```
        if cmd in stopwords:
            break
```

(8) Write a function that applies an arbitrary linear function $f(x) = mx + b$ to a number x. It should take three parameters m, b, x. Then, use this to apply the function $3x + 2$ to the integers $[0, 1, \ldots, 24]$ and the function $x - 7$ to the floats $[0, 0.5, 1, 1.5, 2, 2.5, 3]$.

Answer:
This question didn't specify whether to just compute the values of the function, or to print them. Either is acceptable. Here we print them:

```
def lin_func(m, b, x):
    """Applies a linear function mx + b to a number x."""
    return m*x + b

print("f(x) = 3*x + 2 for x the integers from 0 to 24")
L1 = range(25)
for n in L1:
    print(lin_func(3, 2, n))

print("\nf(x) = x - 7 for x the half-integers from 0 to 3")
L2 = [i/2 for i in range(7)]
for n in L2:
    print(lin_func(1, -7, n))
```

Revision history:

- 2020-09-18 Initial publication