Homework 2: Data Structures in Python

Due Friday, 02.05.2021 at 11:59 PM ET

Goals

This homework has several objectives:
1. Write some basic Python programs.
2. Get familiar with the different data structures available in Python.
3. Leverage the concept of functions to write modular code.

Instructions

In this homework, you need to write two Python functions, one per problem described below. Both of these function definitions are provided to you in homework2.py. testhisto.py and testaddress.py can be used by you to execute your functions in homework2.py. We have provided you with some test cases, you may make your own test case and execute to make sure your code runs properly.

Problem 1

Create a function called histogram that takes as input a dataset data, a lower bound b, an upper bound h, and a number of bins n, and returns a histogram representation of data with n bins between these bounds. More specifically, your function should:

1. Have input arguments histogram(data, n, b, h), expecting data as a list of floats, n as an integer, and b and h as floats.
2. Initialize the histogram hist as a list of n zeros.
3. Calculate the bin width as w = (h-b)/n, so that hist[0] will represent values in the range [b, b + w), hist[1] in the range [b + w, b + 2w), and so on through hist[n-1]. (Remember that [] is inclusive while ) is not!)
4. Ignore any values in data that are less than or equal to b, or greater than or equal to h.
5. Increment hist[i] by 1 for each value in data that
belongs to bin \(i\), i.e., in the range \([b + i \times w, b + (i+1) \times w)\).

6. Return \(hist\).

At the beginning of your function, be sure to check that \(n\) is a positive integer and that \(h \geq b\); if not, your code should just print a message and return an empty list for \(hist\). Your message can be customised by you, but please remember to return an empty list.

For example, typing in

```python
data = [-2, -2.2, 0, 5.6, 8.3, 10.1, 30, 4.4, 1.9, -3.3, 9, 8]
hist = histogram(data, 10, -5, 10)
print(hist)
```

should return

\([0, 2, 1, 1, 1, 0, 1, 1, 2, 1]\)

Some other test cases are:

```python
data = [-4, -3.2, 0, 7.6, 1.0, 2.2, 30, 2.2, 1.9, -8.3, 6, 5]
hist = histogram(data, 10, -5, 10)
print(hist)
```

should return

\([1, 1, 0, 1, 4, 0, 1, 1, 1, 0]\)

and,

```python
data = [2,2,2]
hist = histogram(data, 3, -2, 3)
print(hist)
```
returns

[0, 0, 3]

also,

data = [-1,-1,-1,10,10]
hist = histogram(data, 5, -1, 10)
print(hist)

returns

[0, 0, 0, 0, 0]

Note: Please include all conditions specified in this problem into your code.

**Problem 2**

Create a function called `addressbook` that takes as input two dictionaries, `name_to_phone` and `name_to_address`, and combines them into a single dictionary `address_to_all` that contains the phone number of, and the names of all the people who live at, a given address. Specifically, your function should:

1. Have input arguments `addressbook(name_to_phone, name_to_address)`, expecting `name_to_phone` as a dictionary mapping a name (string) to a home phone number (integer or string), and `name_to_address` as a dictionary mapping a name to an address (string).
2. Create a new dictionary `address_to_all` where the keys are all the addresses contained in `name_to_address`, and the value `address_to_all[address]` for `address` is of the format `([name1,name2,...], phone)`, with `[name1,name2,...]` being the list of names living at
address and phone being the home phone number for address. **Note:** the value we want in this new dictionary is a tuple, where the first element of the tuple is a list of names, and the second element of the tuple is the phone number. (Remember that while a tuple itself is immutable, a list within a tuple can be changed dynamically.)

3. Handle the case where multiple people at the same address have different listed home phone numbers as follows: Keep the first number found, and print warning messages with the names of each person whose number was discarded.

4. Return `address_to_all`.

For example, typing in

```python
name_to_phone = {'alice': 5678982231, 'bob': '111-234-5678', 'christine': 5556412237, 'daniel': '959-201-3198', 'edward': 5678982231}
name_to_address = {'alice': '11 hillview ave', 'bob': '25 arbor way', 'christine': '11 hillview ave', 'daniel': '180 ways court', 'edward': '11 hillview ave'}
address_to_all = addressbook(name_to_phone, name_to_address)
print(address_to_all)
```

should return

```
Warning: christine has a different number for 11 hillview ave than alice. Using the number for alice.
{'11 hillview ave': (["alice", "christine", 'edward'], 5678982231), '25 arbor way': (["bob"], '111-234-5678'), '180 ways court': (["daniel"], '959-201-3198')}
```

also,
name_to_phone = {'alice': 5678982231, 'bob': 5678982231, 'christine': 5678982231, 'daniel': '959-201-3198', 'edward': 5678982231}
name_to_address = {'alice': '25 arbor way', 'bob': '25 arbor way', 'christine': '25 arbor way', 'daniel': '25 arbor way', 'edward': '25 arbor way'}
address_to_all_Stu = addressbook(name_to_phone, name_to_address)
print(address_to_all)

should output

Warning: daniel has a different number for 25 arbor way than alice. Using the number for alice.
{'25 arbor way': (['alice', 'bob', 'christine', 'daniel', 'edward'], 5678982231)}

Your message should match exactly as shown above. If more than one person has a different phone number at the same address then you may print separate messages for each instance.

Note that the specific order you get these elements back may not be the same, because sets and dictionaries do not preserve order. That is OK!

And yes, we know people rarely use home phone numbers anymore, but that doesn't change this problem being a good Python coding exercise! :)

Testing
We have provided two test programs for you, that recreate the examples from above, in testhisto1.py, testhisto2.py, testhisto3.py, testhisto4.py, testaddress1.py and testaddress2.py, which test problems 1 and 2, respectively. Note that these test programs will only work "out of the box" if you have your solution in homework2.py. You may verify your code by running the test programs from the terminal. The concept of importing functions from modules or .py files are being used here.

What to Submit

Put the two functions histogram and addressbook in a single file called homework2.

Once you have a version of this file (that you have committed using git commit and pushed using git push) that you are happy with, you are done! Sit back, relax and enjoy your lectures :)