CSC108: Birthday Dictionary Exercises

Daniel Zingaro
University of Toronto
daniel.zingaro@utoronto.ca

October 2014

1 Birthday Dictionaries (from Fall 2009 exam)

A "birthday month dictionary" is a dictionary of birthday information with the following structure: the keys are string month names, and the values are themselves dictionaries. The keys in each of these sub-dictionaries are integers from 1 to 31 representing days of the month, and the values are lists of strings giving the names of people whose birthdays are on that day. For example, the following birthday month dictionary says that Catherine's birthday is February 13th, Katie's is May 3rd, and both Peter and Ed were born on May 8th:

```
{"February" : {13 : ["Catherine"]}, "May" : {3 : ["Katie"], 8 : ["Peter", "Ed"]}}
```

1.1 Question 1

Complete the following function according to its docstring.

```
def most_covered(bdm):
    '''(dict of {str:dict of {int:list of str}}) -> str

    bdm is a birthday month dictionary with at least one key. Return
    the name of the month that is most "covered" in the dictionary.
    I.e., the month with the most days on which someone has a birthday. If
    there is a tie, return any month with that maximum coverage.
    '''
```

1.1.1 Hints

Before you start answering an exam question like this, make sure you understand the structure of the dictionary. For example, you should be able to create a sample dictionary of the proper form and understand what is done by the following:

- `bdm['February']`
- `bdm['February'][13]`
- `len(bdm['February'])`
- `list(bdm.keys())`
- `list(bdm.values())`
- `list(bdm.values())[0][13]` (careful: the `[0]` part is indexing a list of values, but the order of those values is not known. If the `[0]` accesses a month dictionary with no `13` key, this will be an error)
- `list(list(bdm.values())[0].values())`
- `list(bdm['February'].values())`

Then, decide how you will keep track of the biggest month you have seen so far, and how to remember the number of days of coverage in that month. The question says that the dictionary has at least one key: use this to help you initialize some variables prior to your loop through the dictionary.

1.2 Question 2

Like a birthday month dictionary, a “birthday dictionary” also contains information about birthdays, but in a different format. The keys are people’s names and the values are 2-item lists describing those people’s birthdays. The first item in each list is the name of a month (a str) and the second is the day of that month (an int). For example, the following **equivalent** birthday dictionary has the same content, although in a **different format**, as the birthday month dictionary above.

```
{'Ed': ['May', 8], 'Peter': ['May', 8],
 'Catherine': ['February', 13], 'Katie': ['May', 3]}
```

Complete the following function according to its docstring. Assume that each person’s name in `bdm` is unique (i.e. not repeated in multiple months).

```
def invert_bdm(bdm):
    '''(dict of {str:dict of {int:list of str}}) ->
       dict of {str:list of [str, int]}

    bdm is a birthday month dictionary. Return the equivalent birthday
    dictionary.
    '''
```

1.2.1 Hints

When you have a dictionary in one form and you want to rearrange it into another form, it's likely that the question is asking you to do a **dictionary inversion**. We'll cover how to invert a dictionary in lecture.

In this example, the people's names become the keys, and the months and days become the values. The values are two-element lists. Note that the dictionary will not have any nested dictionaries.