CSC108H: Dictionary Exercises

Dan Zingaro

October 2012

Example 1: List Occurrences

```
def count_occurrences(L):
  '''(list of v) -> dict of {v:int}
  return a dictionary in which the keys are
  the items in L and their associated values
  are integers denoting the number of times the item is
  contained in L.

  >>> count_occurrences([8, 9, 8, 8, 9])
  {8:3, 9:2}
  '''
```

Example 2: Folding Dictionaries

```
def fold(d1, d2):
  '''(dict, dict) -> dict
  Return a new dictionary that contains all (b, c)
  such that (a, b) is in d1 and (a, c) is in d2.

  >>> fold({1:4, 9:10}, {4:5})
  {4:5}
  '''
```

Is the folded dictionary **guaranteed** to be unique?

Example 3: Sum of List Values

```
def combine(d1, d2):
  '''(dict of {v:list of int}, dict of {w:list of int}) ->
     dict of {x:int}
  Return the dictionary where each key is a key
  that is in both d1 and d2.
  The value associated with each key in the new
  dictionary is the sum of all the integers associated
  with that key in d1 and d2.

  >>> combine({1:[2], 4:[5, 6]}, {4:[8]}
  {4:19}
  '''
```

Sparse Matrices

- A sparse matrix is a matrix whose entries are almost all zero

$$\begin{pmatrix} 0 & 0 & 4 \\ 0 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix}$$

- Storing sparse matrices as lists of lists can waste a lot of memory
- Alternative: use a dictionary whose keys are (row, column) tuples and whose values are the values at those coordinates
- e.g. for the above: `{(0, 2):4, (2,1):3}`

Sparse Matrices...

To add two matrices, we add their corresponding components

$$\begin{pmatrix} 0 & 0 & 4 \\ 0 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 1 \\ 0 & 12 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0 & 5 \\ 0 & 12 & 0 \\ 0 & 3 & 0 \end{pmatrix}$$

Write a function that takes two sparse matrices stored as dictionaries and returns a new dictionary representing their sum