1 Ranking web pages in search results

Before Google, web search engines like Altavista ranked results based on the text of the web page. One of the insights of the founders of Google was to come up with a way of assigning a quality score to each page on the web and use these scores to have higher quality web pages appear higher in the search results. The resulting measure PageRank is based on the idea of a "random surfer," described below. Before getting to that, we consider a few alternative measures.

Popularity Here's one candidate measure of quality: popularity. A page is high quality if many other pages link to it.

1. Write a function `popularity` that takes in a web graph and returns a dictionary that maps each web page to its popularity. The popularity of a web page is the number of *incoming* links.

   A web graph is represented as a dictionary that maps page names to the list of outgoing links on that page.

   Example:

```
>>> web_graph = {'a.html': ['b.html', 'c.html'],
                 'b.html': ['c.html'], 'c.html': []}
>>> popularity(web_graph)
{'a.html': 0, 'b.html': 1, 'c.html': 2}
```

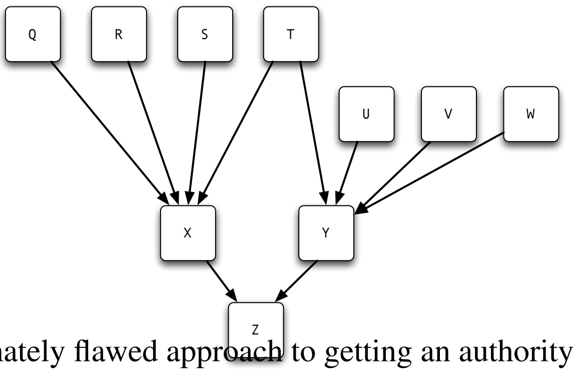
Solution:

```
def popularity(web_graph):
    '''(dict of str: list of str) -> dict of str:int
    Returns a dictionary that reports the number of
    incoming links for each page in web_graph.
    >>> web_graph = {'a.html': ['b.html', 'c.html'],
                     'b.html': ['c.html'],
                     'c.html': []}
    >>> popularity(web_graph)
    {'a.html': 0, 'b.html': 1, 'c.html': 2}
    '''
    inlinks = {}
    for page in web_graph:
        inlinks[page] = 0

    for page in web_graph:
        for linked_page in web_graph[page]:
            inlinks[linked_page] += 1
    return inlinks
```

Authority Popularity treats all links equally, but perhaps some links are worth more than others. Consider the following web graph. Even though Z has only two incoming links, they are from X and Y, the most popular pages. Perhaps links from popular pages should be given more weight.

We might propose to rank pages by "authority." The idea is that we view links as endorsements of authority. A page can have high authority if it has lots of incoming links, but it can also have authority if it is linked to by pages that are themselves authoritative. In this graph, Z might be considered the most authoritative.



Here's one intuitive but ultimately flawed approach to getting an authority score:

- initialize each page to have an authority score of 1
- update the authority of page p to the *sum* of authority score of the pages that point to p

This is intuitively appealing but runs into trouble when the web graph has a cycle (e.g., A points to B, B points to C, and C points back to A).

Random surfer Imagine someone who surfs the web randomly. The random surfer moves around the web by just randomly clicking links, never hitting the back button. However, occasionally the surfer gets bored and just jumps to a random page (i.e., types a new address into the box at the top of the browser). It turns out that if we simply count how often the random surfer lands on a page, it corresponds to a pretty good measure that captures both popularity and authority. It also avoids the issue of cycles that arose with the authority measure.

2. Write pseudocode (i.e., a hybrid between English and python) that simulates the random surfer over a web graph.

Solution:

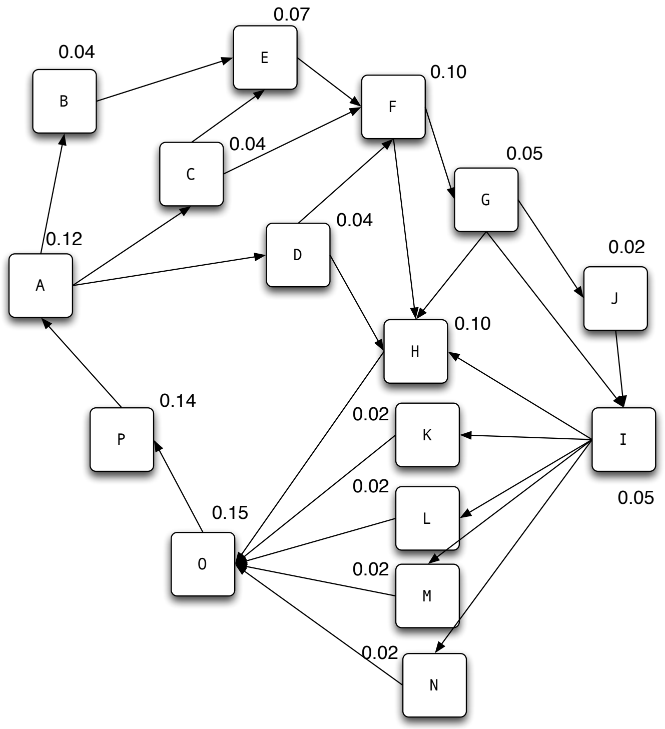
```
def random_surfer_simulation(web_graph, p, num_sims):
    current_page = choose a random page from web

    repeat num_sims times:
        r = random.random()
        if r <= p:
            current_page = choose a random page from web
        else:
            links = get links of current_page
            current_page = choose a random page from links

        increase number of visits to current_page by 1
        visits[current_page] += 1

    convert visits into ranks by dividing by total number of visits
```

Authority scores from a random surfer simulation The figure below shows the web graph for the homework and the “authority” scores that result from a simulation of the random surfer.



Adapted from Nine Algorithms that Changed the Future, MacCormick

The random surfer score captures popularity. If page p has many incoming links, it is likely to be visited often because if the random surfer ends up on *any* of the pages that link to p, the surfer has a high chance of landing on p next. For example, page O is popular and it has a high score.

The random surfer also captures the “authority” idea. For example, page A and J each have one incoming link but A’s incoming link is from a more “authoritative” source, giving A a much higher score than J.