Dalziel2015_solution

January 7, 2016

1 Solution of Dalziel et al. 2015

1.1 Write a function that takes as arguments the name of a city and the index of a biweek (from 1 to 26). The function should return the number of cases of measles for that city and biweekly period, for all the available years.

First, we are going to build the pieces that go into the function. Once we're done, we're going to write the function itself. For testing, we choose `BALTIMORE` as the city and `3` as the biweek.

```
In [1]: city = 'BALTIMORE'
```

```
In [2]: biweek = 3
```

Open the file using the `csv` package:

```
In [3]: import csv
        with open('../data/Dalziel2015_data.csv') as csvfile:
            reader = csv.DictReader(csvfile)
            print(next(reader))
```

```
{'biweek': '1', 'loc': 'BALTIMORE', 'year': '1906', 'pop': '526822.1365', 'cases': 'NA'}
```

As you can see, this reads each row in the file an converts it into a dictionary, using the header to name the keys. This makes it very simple to filter the desired records:

```
In [4]: with open('../data/Dalziel2015_data.csv') as csvfile:
            reader = csv.DictReader(csvfile)
            for row in reader:
                if row['loc'] == city and int(row['biweek']) == biweek:
                    print(row['year'], row['cases'])
```

```
1906 NA
1907 129
1908 70
1909 177
1910 44
1911 261
1912 14
1913 163
1914 17
1915 9
1916 402
1917 33
1918 240
1919 25
```

```
1920 270
1921 74
1922 215
1923 160
1924 84
1925 13
1926 2341
1927 5
1928 878
1929 4
1930 4
1931 592
1932 5
1933 3
1934 213
1935 29
1936 31
1937 574
1938 12
1939 1938
1940 4
1941 8
1942 388
1943 23
1944 712
1945 13
1946 84
1947 9
1948 5
```

Now that we have the code ready, we can write a function:

```
In [5]: import csv

        def get_cases_city_biweek(city = "BALTIMORE", biweek = 5):
            with open('../data/Dalziel2015_data.csv') as csvfile:
                reader = csv.DictReader(csvfile)
                for row in reader:
                    if row['loc'] == city and int(row['biweek']) == biweek:
                        print(row['year'], row['cases'])
```

We can call the function using another city/biweek, for example `BOSTON` and `7`:

```
In [6]: get_cases_city_biweek('BOSTON', 7)
```

```
1906 NA
1907 26
1908 414
1909 150
1910 276
1911 332
1912 325
1913 497
1914 188
1915 512
```

```
1916 293
1917 360
1918 489
1919 21
1920 456
1921 286
1922 400
1923 323
1924 347
1925 610
1926 341
1927 159
1928 825
1929 31
1930 876
1931 223
1932 129
1933 278
1934 773
1935 79
1936 761
1937 33
1938 441
1939 322
1940 149
1941 554
1942 371
1943 348
1944 291
1945 216
1946 779
1947 123
1948 826
```

1.2 Write a function that returns all the cities in the dataset.

There are different ways to do this. Here, the strategy is to construct a `set`, so that cities are not repeated:

```
In [7]: def get_all_cities():
            all_cities = set([])
            with open('../data/Dalziel2015_data.csv') as csvfile:
                reader = csv.DictReader(csvfile)
                for row in reader:
                    all_cities.add(row['loc'])
            return(all_cities)
```

Now invoke the function:

```
In [8]: get_all_cities()
```

```
Out[8]: {'BALTIMORE',
         'BOSTON',
         'BRIDGEPORT',
         'BUFFALO',
         'CHICAGO',
```

```
        'CINCINNATI',
        'CLEVELAND',
        'COLUMBUS',
        'DENVER',
        'DETROIT',
        'DULUTH',
        'FALL RIVER',
        'GRAND RAPIDS',
        'HARTFORD',
        'INDIANAPOLIS',
        'KANSAS CITY',
        'LOS ANGELES',
        'MILWAUKEE',
        'MINNEAPOLIS',
        'NASHVILLE',
        'NEW HAVEN',
        'NEW ORLEANS',
        'NEW YORK',
        'NEWARK',
        'PHILADELPHIA',
        'PITTSBURGH',
        'PROVIDENCE',
        'READING.US',
        'RICHMOND',
        'ROCHESTER',
        'SALT LAKE CITY',
        'SAN FRANCISCO',
        'SEATTLE',
        'SPOKANE',
        'SPRINGFIELD',
        'ST LOUIS',
        'TOLEDO',
        'TRENTON',
        'WASHINGTON',
        'WORCESTER'}
```

In [ ]: