BeeChecklist_solution

January 20, 2016

0.1 Solution of Bee Checklist

First of all, we import the two modules we'll need to read the `csv` file, and to use regular expressions:

```
In [1]: import csv
        import re
```

Then, we read the file, and store the columns `Scientific Name` and `Taxon Author` in two lists:

```
In [2]: with open('../data/bee_list.txt') as f:
            csvr = csv.DictReader(f, delimiter = '\t')
            species = []
            authors = []
            for r in csvr:
                species.append(r['Scientific Name'])
                authors.append(r['Taxon Author'])
```

How many species?

```
In [3]: len(species)
```

```
Out[3]: 19508
```

```
In [4]: len(authors)
```

```
Out[4]: 19508
```

Pick one of the `authors` element to use for testing. Choose one that is quite complicated, such as the 38th element:

```
In [5]: au = authors[37]
```

```
In [6]: au
```

```
Out[6]: 'Tadauchi, Hirashima & Matsumura, 1987'
```

Now we need to build a regular expression. After some twiddling, you should end up with something like this, which captures the authors in one group, and the year in another group:

```
In [7]: my_reg = re.compile(r'\(?([\w\s,\.\-\&]*),\s(\d{4})\)?')
        # Translation
        # \(? -> open parenthesis (or not)
        # ([\w\s,\.\-\&]+) -> the first group is the list of authors
        #                     which can contain \w (word character)
        #                     \s (space) \. (dot) \- (dash) \& (ampersand)
        # ,\s -> followed by comma and space
        # (\d{4}) -> the second group is the year, 4 digits
        #  \)? -> potentially, close parenthesis
```

Test the expression

```
In [8]: re.findall(my_reg,au)
```

```
Out[8]: [('Tadauchi, Hirashima & Matsumura', '1987')]
```

Now we write a function that uses the regular expression to extract an author list (useful when there are multiple authors), and the year

```
In [9]: def extract_list_au_year(au):
            tmp = re.match(my_reg, au)
            authorlist = tmp.group(1)
            year = tmp.group(2)
            # split authors into a list using re.split
            authorlist = re.split(', | \& ', authorlist)
            # Translation: either separate using ', ' or ' & '
            return [authorlist, year]
```

Let's see the output of this function:

```
In [10]: extract_list_au_year(au)
```

```
Out[10]: [['Tadauchi', 'Hirashima', 'Matsumura'], '1987']
```

Finally, let's build two dictionaries: - one tracking the number of times each year is mentioned in the database; - one traking the number of times each author is mentioned

```
In [11]: dict_years = {}
         dict_authors = {}
         for au in authors:
             tmp = extract_list_au_year(au)
             for aunum in tmp[0]:
                 if aunum in dict_authors.keys():
                     dict_authors[aunum] = dict_authors[aunum] + 1
                 else:
                     dict_authors[aunum] =  1
             if tmp[1] in dict_years.keys():
                 dict_years[tmp[1]] = dict_years[tmp[1]] + 1
             else:
                 dict_years[tmp[1]] =  1
```

For example, these are all the authors:

```
In [12]: dict_authors
```

```
Out[12]: {'Abe': 1,
          'Abramovich': 2,
          'Albuquerque': 8,
          'Alfken': 149,
          'Almeida': 6,
          'Anjos-Silva': 1,
          'Arnold': 1,
          'Ascher': 1,
          'Ashmead': 30,
          'Astafurova': 2,
          'Atanassov': 1,
          'Atwood': 6,
```

```
'Audinet-Serville': 11,
'Aurivillius': 1,
'Ayala': 22,
'Azevedo': 2,
'Baker': 44,
'Banaszak': 1,
'Batra': 1,
'Bembé': 1,
'Bennett': 1,
'Benoist': 164,
'Benzi': 1,
'Bertoni': 7,
'Bingham': 47,
'Bischoff': 10,
'Blackburn': 3,
'Blanchard': 4,
'Blüthgen': 315,
'Bohart': 47,
'Boongird': 1,
'Borges': 1,
'Bouseman': 8,
'Bradley': 1,
'Bramson': 1,
'Brauns': 35,
'Bridwell': 16,
'Broemeling': 5,
'Brooks': 103,
'Brues': 2,
'Brullé': 33,
'Brèthes': 42,
'Burmeister': 5,
'Buysson': 6,
'Bytinski-Salz': 1,
'Bär': 1,
'Cabezas': 15,
'Camargo': 75,
'Cameron': 158,
'Cardale': 1,
'Casad': 7,
'Castro': 1,
'Cheesman': 30,
'Chevrier': 1,
'Chiappa': 3,
'Christ': 4,
'Cockerell': 3394,
'Coelho': 5,
'Compagnucci': 10,
'Cooper': 3,
'Costa': 6,
'Crawford': 88,
'Cresson': 437,
'Cross': 2,
'Cunha': 1,
'Cure': 11,
```

```
'Curtis': 4,
'Dahlbom': 2,
'Dalla Torre': 65,
'Daly': 31,
'Danforth': 4,
'Darchen': 2,
'Dathe': 38,
'Davies': 11,
'Davydova': 1,
'Dawut': 1,
'De Geer': 1,
'De Stefani': 1,
'DeGeer': 6,
'Dewitz': 1,
'Deyrup': 1,
'Doering': 1,
'Dominique': 2,
'Donovan': 24,
'Dours': 62,
'Dover': 3,
'Dressler': 36,
'Drury': 2,
'Dubitzki': 1,
'Dubitzky': 12,
'Ducke': 92,
'Dufour': 4,
'Dumeril': 1,
'Dunning': 3,
'Durante': 4,
'Dusmet y Alonso': 33,
'E. A. B. Almeida': 5,
'Eardley': 93,
'Ebmer': 215,
'Ehrenfeld': 3,
'Eickwort': 7,
'Ellis': 23,
'Enderlein': 19,
'Engel': 89,
'Erichson': 23,
'Erlandsson': 1,
'Evans': 4,
'Eversmann': 41,
'Exley': 231,
'Fabricius': 138,
'Fan': 13,
'Fedtschenko': 29,
'Ferton': 6,
'Fonscolombe': 3,
'Forster': 1,
'Fourcroy': 1,
'Fowler': 7,
'Fox': 25,
'Franck': 1,
'Franklin': 7,
```

```
'Frey-Gessner': 1,
'Friese': 1330,
'Frison': 13,
'Fritz': 15,
'Förster': 21,
'Gaiani': 3,
'Genaro': 16,
'Geoffroy': 2,
'Germar': 2,
'Gerstäcker': 55,
'Gibbs': 1,
'Giordani Soika': 1,
'Giraud': 12,
'Gistel': 3,
'Gmelin': 1,
'Godínez-García': 1,
'Gonzalez': 19,
'González': 3,
'González-Vaquero': 3,
'Gonçalves': 4,
'Gorski': 1,
'Graenicher': 9,
'Graf': 2,
'Greene': 3,
'Gribodo': 75,
'Grigarick': 2,
'Griswold': 50,
'Grünwaldt': 7,
'Guiglia': 2,
'Guilding': 1,
'Gupta': 16,
'Gusenleitner': 39,
'Gussakovsky': 14,
'Guérin-Méneville': 19,
'H. S. Smith': 7,
'Haeseler': 1,
'Hagens': 10,
'Haliday': 7,
'Handlirsch': 8,
'Haneda': 1,
'Harris': 1,
'Harter-Marques': 2,
'Hazir': 1,
'He': 2,
'Hedicke': 24,
'Heinrich': 3,
'Hensen': 1,
'Herbst': 11,
'Herrera': 3,
'Herrich-Schäffer': 8,
'Herrmann': 1,
'Hicks': 2,
'Hill': 1,
'Hinojosa-Díaz': 8,
```

```
'Hirashima': 210,
'Hoffmannsegg': 1,
'Holmberg': 181,
'Houston': 50,
'Huang': 1,
'Huard': 2,
'Hurd': 36,
'Ikudome': 8,
'Illiger': 9,
'Imhoff': 8,
'Inoue': 3,
'Itino': 1,
'Jaeger': 1,
'Janjic': 1,
'Janvier': 2,
'Jaycox': 1,
'Jensen-Haarup': 2,
'Jobiraj': 1,
'Jurine': 4,
'Jörgensen': 38,
'Kato': 1,
'Kerr': 3,
'Kim': 2,
'Kimsey': 10,
'King': 9,
'Kirby': 74,
'Klein': 2,
'Klug': 35,
'Knerer': 5,
'Kocourek': 4,
'Kohl': 18,
'Kokujev': 1,
'Kreichbaumer': 1,
'Kriechbaumer': 7,
'Krombein': 11,
'Kuhlmann': 65,
'LaBerge': 192,
'LaRoche': 3,
'Labougle': 1,
'Laboulbene': 1,
'Lanham': 4,
'Larkin': 6,
'Laroca': 1,
'Latreille': 35,
'Le Goff': 1,
'LeVeque': 10,
'Lebedev': 9,
'Lelej': 2,
'Lepeletier': 185,
'Leys': 2,
'Lieftinck': 108,
'Linnaeus': 39,
'Linné': 3,
'Linsley': 59,
```

```
'Lobo Segura': 1,
'Lovell': 17,
'Lozinski': 2,
'Lucas': 9,
'Lucas de Oliveira': 1,
'Lucia': 2,
'M. C. de Almeida': 1,
'Maa': 22,
'MacSwain': 16,
'Mackie': 2,
'Maeta': 9,
'Magnacca': 10,
'Magretti': 18,
'Maidl': 16,
'Malloch': 3,
'Marchi': 9,
'Mariskovskaya': 1,
'Masi': 1,
'Matsumura': 19,
'Mavromoustakis': 100,
'Maynard': 16,
'Mazzucco': 1,
'McGinley': 24,
'Meade-Waldo': 45,
'Melo': 23,
'Metz': 16,
'Meunier': 5,
'Meyer': 31,
'Michener': 455,
'Michez': 14,
'Milliron': 1,
'Mitai': 3,
'Mitchell': 270,
'Miyanaga': 2,
'Moalif': 4,
'Mocsáry': 67,
'Moldenke': 37,
'Morawitz': 513,
'Morice': 11,
'Motschulski': 1,
'Moure': 417,
'Munakata': 1,
'Munzinger': 2,
'Murao': 4,
'Müller': 4,
'Nagase': 3,
'Narendran': 1,
'Neff': 5,
'Nemésio': 5,
'Neves': 1,
'Niu': 1,
'Nobile': 7,
'Noble': 1,
'Noskiewicz': 75,
```

```
'Nurse': 53,
'Nylander': 37,
'Ogloblin': 7,
'Oliveira': 8,
'Olivier': 14,
'Ordway': 3,
'Ornosa': 1,
'Ortiz': 1,
'Ortiz-Sánchez': 1,
'Ospina-Torres': 2,
'Osytshnjuk': 74,
'Packard': 8,
'Packer': 29,
'Pallas': 4,
'Panzer': 29,
'Parker': 15,
'Pasteels': 197,
'Patiny': 24,
'Patton': 5,
'Pauly': 175,
'Pedro': 25,
'Perkins': 65,
'Perris': 4,
'Perty': 5,
'Pesenko': 56,
'Peters': 16,
'Philippi': 2,
'Pittioni': 7,
'Pohl': 1,
'Ponomareva': 2,
'Popov': 86,
'Porter': 4,
'Priesner': 20,
'Proschchalykin': 1,
'Proshchalykin': 1,
'Provancher': 25,
'Puls': 1,
'Pérez': 309,
'Quilis': 2,
'R. P. Urban': 5,
'Radoszkowski': 113,
'Rahman': 1,
'Ramos': 1,
'Ramírez': 2,
'Rasmont': 4,
'Rasmussen': 1,
'Raw': 7,
'Rayment': 209,
'Rebmann': 18,
'Rebêlo': 5,
'Reed': 5,
'Reyes': 21,
'Ribble': 31,
'Richards': 7,
```

```
'Rightmyer': 40,
'Risch': 23,
'Ritsema': 16,
'Roberts': 32,
'Robertson': 135,
'Robinaeau': 1,
'Rodeck': 1,
'Rodriguez': 2,
'Rodríguez': 3,
'Rohwer': 2,
'Roig-Alsina': 71,
'Rojas': 11,
'Romand': 1,
'Romankova': 4,
'Rossi': 11,
'Roubik': 13,
'Rozen': 42,
'Rudow': 2,
'Ruiz': 16,
'Rust': 3,
'Ruz': 45,
'S. Lee': 1,
'Sakagami': 38,
'Sandhouse': 78,
'Sandino-Franco': 1,
'Saunders': 22,
'Saussure': 28,
'Say': 35,
'Schenck': 41,
'Scheuchl': 10,
'Schilling': 1,
'Schletterer': 23,
'Schlindwein': 15,
'Schmidt': 1,
'Schmiedeknecht': 50,
'Schrank': 2,
'Schrottky': 246,
'Schuberth': 1,
'Schulten': 7,
'Schulthess': 1,
'Schulthess-Rechberg': 1,
'Schulz': 13,
'Schummel': 1,
'Schwammberger': 8,
'Schwarz': 170,
'Schwenninger': 1,
'Schwimmer': 1,
'Schönherr': 1,
'Schönitzer': 2,
'Scopoli': 8,
'Seabra': 12,
'Seidl': 1,
'Sepúlveda': 1,
'Shanks': 15,
```

```
'Sharma': 6,
'Shinn': 25,
'Shiokawa': 4,
'Shrottky': 1,
'Sichel': 22,
'Sickmann': 1,
'Sielfeld': 1,
'Silveira': 7,
'Silveira et al.': 1,
'Silvestri': 2,
'Simlote': 2,
'Sitdikov': 5,
'Sivik': 1,
'Skorikov': 22,
'Skov': 1,
'Sladen': 3,
'Smith': 943,
'Smith-Pardo': 10,
'Snelling': 72,
'Sonan': 1,
'Sparre-Schneider': 1,
'Spinola': 108,
'Stadelmann': 5,
'Stage': 4,
'Standfuss': 4,
'Stanek': 5,
'Stange': 3,
'Steiner': 13,
'Stephen': 12,
'Stephens': 1,
'Stevens': 4,
'Stoeckhert': 10,
'Strand': 249,
'Sumner': 1,
'Svensson': 1,
'Swederus': 2,
'Swenk': 62,
'Syed': 2,
'Tadauchi': 76,
'Takahashi': 1,
'Tamasana': 1,
'Tapia': 4,
'Taschenberg': 2,
'Terzo': 12,
'Teunissen': 1,
'Tewari': 1,
'Theunert': 1,
'Thiele': 1,
'Thomson': 9,
'Thorp': 19,
'Tierney': 1,
'Timberlake': 864,
'Timmermann': 1,
'Titus': 10,
```

```
'Tkalcu': 86,
'Toro': 140,
'Trucco Aleman': 5,
'Truylio': 1,
'Tsuneki': 74,
'Tucker': 1,
'Turrisi': 7,
'Uchida': 3,
'Urban': 282,
'Vachal': 557,
'Vecht': 1,
'Vergara': 1,
'Verhoeff': 1,
'Viana': 2,
'Viereck': 130,
'Vivallo': 3,
'Vogel': 1,
'Vogt': 5,
'Vollenhoven': 1,
'Vélez': 1,
'W. F. Kirby': 13,
'Wahlberg': 1,
'Walckenaer': 1,
'Walker': 112,
'Wang': 1,
'Warncke': 507,
'Wcislo': 3,
'Wesmael': 1,
'Westrich': 3,
'Westwood': 11,
'White': 4,
'Whitehead': 16,
'Wickwar': 1,
'Wiedemann': 1,
'Wille': 3,
'Willis': 3,
'Wittman': 1,
'Wolcott': 1,
'Wu': 213,
'Xu': 37,
'Yanega': 6,
'Yasumatsu': 53,
'Yáñez-Ordóñez': 3,
'Zanella': 4,
'Zavortink': 7,
'Zetterstedt': 5,
'de Beaumont': 1,
'de Gaulle': 1,
'de Villers': 1,
'van Achterberg': 1,
'van der Vecht': 12,
'van der Zanden': 53,
'von Schulthess': 1}
```

0.1.1 What is the name of the author with most entries in the database?

We use the following strategy: - we find the maximum value in the dictionary - we use the function index to find to which entry is it associated - we find the corresponding author

```
In [13]: max_value_author = max(dict_authors.values())
         max_value_author
```

```
Out[13]: 3394
```

```
In [14]: which_index = list(dict_authors.values()).index(max_value_author)
         which_index
```

```
Out[14]: 97
```

An the winner is:

```
In [15]: list(dict_authors.keys())[which_index]
```

```
Out[15]: 'Cockerell'
```

0.1.2 Which year of publication is most represented in the database?

We use the same strategy to find that the golden year of bee publication is:

```
In [16]: max_value_year = max(dict_years.values())
         which_index = list(dict_years.values()).index(max_value_year)
         list(dict_years.keys())[which_index]
```

```
Out[16]: '1903'
```