

## Assignment 1

In this assignment, you'll be working with messy medical data and using regex to extract relevant information from the data.

Each line of the `dates.txt` file corresponds to a medical note. Each note has a date that needs to be extracted, but each date is encoded in one of many formats.

The goal of this assignment is to correctly identify all of the different date variants encoded in this dataset and to properly normalize and sort the dates.

Here is a list of some of the variants you might encounter in this dataset:

- 04/20/2009; 04/20/09; 4/20/09; 4/3/09
- Mar-20-2009; Mar 20, 2009; March 20, 2009; Mar. 20, 2009; Mar 20 2009;
- 20 Mar 2009; 20 March 2009; 20 Mar. 2009; 20 March, 2009
- Mar 20th, 2009; Mar 21st, 2009; Mar 22nd, 2009
- Feb 2009; Sep 2009; Oct 2010
- 6/2008; 12/2009
- 2009; 2010

Once you have extracted these date patterns from the text, the next step is to sort them in ascending chronological order according to the following rules:

- Assume all dates in `xx/xx/xx` format are `mm/dd/yy`
- Assume all dates where year is encoded in only two digits are years from the 1900's (e.g. 1/5/89 is January 5th, 1989)
- If the day is missing (e.g. 9/2009), assume it is the first day of the month (e.g. September 1, 2009).
- If the month is missing (e.g. 2010), assume it is the first of January of that year (e.g. January 1, 2010).
- Watch out for potential typos as this is a raw, real-life derived dataset.

With these rules in mind, find the correct date in each note and return a pandas Series in chronological order of the original Series' indices.

For example if the original series was this:

```
0    1999
1    2010
2    1978
3    2015
4    1985
```

Your function should return this:

```
0    2
1    4
2    0
3    1
4    3
```

Your score will be calculated using [Kendall's tau](#), a correlation measure for ordinal data.

*This function should return a Series of length 500 and dtype int.*

```
In [2]: import pandas as pd

doc = []
with open('dates.txt') as file:
    for line in file:
        doc.append(line)

df = pd.Series(doc)
df.head(10)

Out[2]: 0      03/25/93 Total time of visit (in minutes):\n
1      6/18/85 Primary Care Doctor:\n
2      sshe plans to move as of 7/8/71 In-Home Servic...\n
3      7 on 9/27/75 Audit C Score Current:\n
4      2/6/96 sleep studyPain Treatment Pain Level (N...\n
5      .Per 7/06/79 Movement D/O note:\n
6      4, 5/18/78 Patient's thoughts about current su...\n
7      10/24/89 CPT Code: 90801 - Psychiatric Diagnos...\n
8      3/7/86 SOS-10 Total Score:\n
9      (4/10/71)Score-1Audit C Score Current:\n
dtype: object

In [3]: def date_sorter():

    # Your code here
    global df
    dates_extracted = df.str.extractall(r'(?P<origin>(?(P<month>\d?\d)[/|-](?(P<day>\d?\d)[/|-](?(P<year>\d{4})|))')
    index_left = ~df.index.isin([x[0] for x in dates_extracted.index])
    dates_extracted = dates_extracted.append(df[index_left].str.extractall(r'(?P<origin>(?(P<month>\d?\d)[/|-](?(P<day>{
0-2}?[0-9])|([3][01]))[/|-](?(P<year>\d{2})|))'))
    index_left = ~df.index.isin([x[0] for x in dates_extracted.index])
    del dates_extracted[3]
    del dates_extracted[4]
    dates_extracted = dates_extracted.append(df[index_left].str.extractall(r'(?P<origin>(?(P<day>\d?\d) ?(?(P<month>[a-z
A-Z]{3,})\.\.? ?(?(P<year>\d{4})|))'))
    index_left = ~df.index.isin([x[0] for x in dates_extracted.index])
    dates_extracted = dates_extracted.append(df[index_left].str.extractall(r'(?P<origin>(?(P<month>[a-zA-Z]{3,})\.\.? ?
(P<day>\d\d?)(th|nd|st)?,?~? ?(?(P<year>\d{4})|))'))
    del dates_extracted[3]
    index_left = ~df.index.isin([x[0] for x in dates_extracted.index])

    # Without day
    dates_without_day = df[index_left].str.extractall('(?P<origin>(?(P<month>[A-Z][a-z]{2,})\.\.? ?(?(P<year>\d{4})|))')
    dates_without_day = dates_without_day.append(df[index_left].str.extractall(r'(?P<origin>(?(P<month>\d\d?)/(?(P<year>
\d{4})|))'))
    dates_without_day['day'] = 1
    dates_extracted = dates_extracted.append(dates_without_day)
    index_left = ~df.index.isin([x[0] for x in dates_extracted.index])

    # Only year
    dates_only_year = df[index_left].str.extractall(r'(?P<origin>(?(P<year>\d{4})|))')
    dates_only_year['day'] = 1
    dates_only_year['month'] = 1
    dates_extracted = dates_extracted.append(dates_only_year)
    index_left = ~df.index.isin([x[0] for x in dates_extracted.index])

    # Year
    dates_extracted['year'] = dates_extracted['year'].apply(lambda x: '19' + x if len(x) == 2 else x)
    dates_extracted['year'] = dates_extracted['year'].apply(lambda x: str(x))

    # Month
    else x)
    month_dict = dict({'September': 9, 'Mar': 3, 'November': 11, 'Jul': 7, 'January': 1, 'December': 12,
        'Feb': 2, 'May': 5, 'Aug': 8, 'Jun': 6, 'Sep': 9, 'Oct': 10, 'June': 6, 'March': 3,
        'February': 2, 'Dec': 12, 'Apr': 4, 'Jan': 1, 'Janaury': 1, 'August': 8, 'October': 10,
        'July': 7, 'Since': 1, 'Nov': 11, 'April': 4, 'Decemeber': 12, 'Age': 8})
    dates_extracted.replace({"month": month_dict}, inplace=True)
    dates_extracted['month'] = dates_extracted['month'].apply(lambda x: str(x))

    # Day
    dates_extracted['day'] = dates_extracted['day'].apply(lambda x: str(x))

    # Cleaned date
    dates_extracted['date'] = dates_extracted['month'] + '/' + dates_extracted['day'] + '/' + dates_extracted['year']
    dates_extracted['date'] = pd.to_datetime(dates_extracted['date'])

    dates_extracted.sort_values(by='date', inplace=True)
    df1 = pd.Series(list(dates_extracted.index.labels[0]))

    return df1

date_sorter()

Out[3]: 0      9
1     84
2      2
3     53
4     28
5    474
6    153
7     13
8    129
9     98
10    111
11   225
12     31
13   171
14   191
15   486
16   335
17   415
18     36
19   405
20   323
21   422
22   375
23   380
24   345
25     57
26   481
27   436
28   104
29   299
...
470   220
471   243
472   208
473   139
474   320
475   383
476   286
477   244
478   480
479   431
480   279
481   198
482   381
483   463
484   366
485   439
486   255
487   401
488   475
489   257
490   152
491   235
492   464
493   253
494   231
495   427
496   141
497   186
498   161
499   413
Length: 500, dtype: int64

In [ ]:
```