1. Significant earthquakes since 2150 B.C.

The Significant Earthquake Database contains information on destructive earthquakes from 2150 B.C. to the present. Select all columns and download the entire significant earthquake data file in `.tsv` format by clicking the `Download TSV File` button. Click the variable name for more information. Read the file (e.g., `earthquakes-2021-10-13_13-22-50_+0800.tsv`) as an object and name it `Sig_Eqs`.

1.1 [5 points] Compute the total number of deaths caused by earthquakes since 2150 B.C. in each country, and then print the top ten countries along with the total number of deaths.

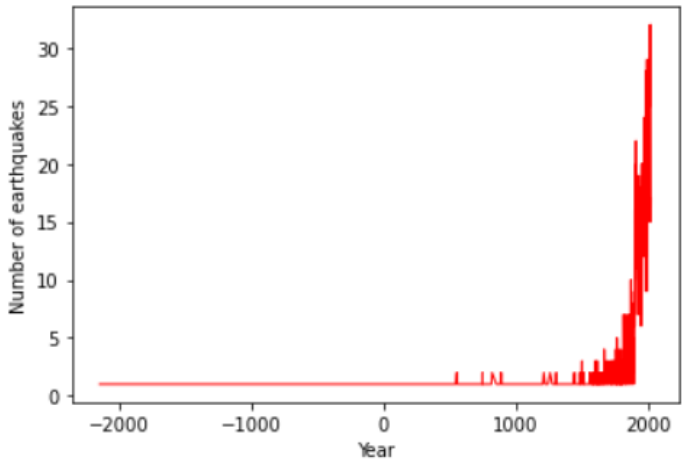
1.2 [10 points] Compute the total number of earthquakes with magnitude larger than `6.0` (use column `Mag` as the magnitude) worldwide each year, and then plot the time series. Do you observe any trend? Explain why or why not?

1.3 [10 points] Write a function `CountEq_LargestEq` that returns both (1) the total number of earthquakes since 2150 B.C. in a given country AND (2) the date of the largest earthquake ever happened in this country. Apply `CountEq_LargestEq` to every country in the file, report your results in a descending order.

```
#PS2_1.1
S1=Sig_Eqs.groupby(['Country']).['Deaths'].count()
S1.sort_values(ascending=False).head(10)
```

```
Country
CHINA          277
IRAN           187
TURKEY         164
INDONESIA      134
ITALY          119
JAPAN          108
PERU            86
GREECE          77
TAIWAN          66
PHILIPPINES     60
Name: Deaths, dtype: int64
```

```
#PS2_1.2 plot the time series
import matplotlib.pyplot as plt
plt.xlabel("Year")
plt.ylabel("Number of earthquakes")
plt.plot(S2, color='red', linewidth=1.0)
plt.show()
```



```
#PS2_1.3 apply to every country
new=pd.DataFrame(columns=['Country','Mag','date'])
country=[]
for (Country),group in Sig_Eqs[Sig_Eqs['Mag']>0].groupby('Country'):
    country.append(Country)

x=0
for i in country:
    new.loc[str(x)]=[i,CountEq_LargestEq(i)[1],CountEq_LargestEq(i)[2]]
    x+=1
new=new.sort_values('Mag',ascending=False)
new[["Country","Mag"]]
```

	Country	Mag
25	CHINA	610
66	JAPAN	409
60	INDONESIA	401
61	IRAN	380
131	TURKEY	330
...	...	...
77	MADAGASCAR	1
94	NORWAY	1
97	PALAU	1
110	SIERRA LEONE	1
145	ZAMBIA	1

146 rows × 2 columns

Note: For question 1.3, I don't know how to write codes for running every country, then my classmate Mai Zelin told me how to do it.

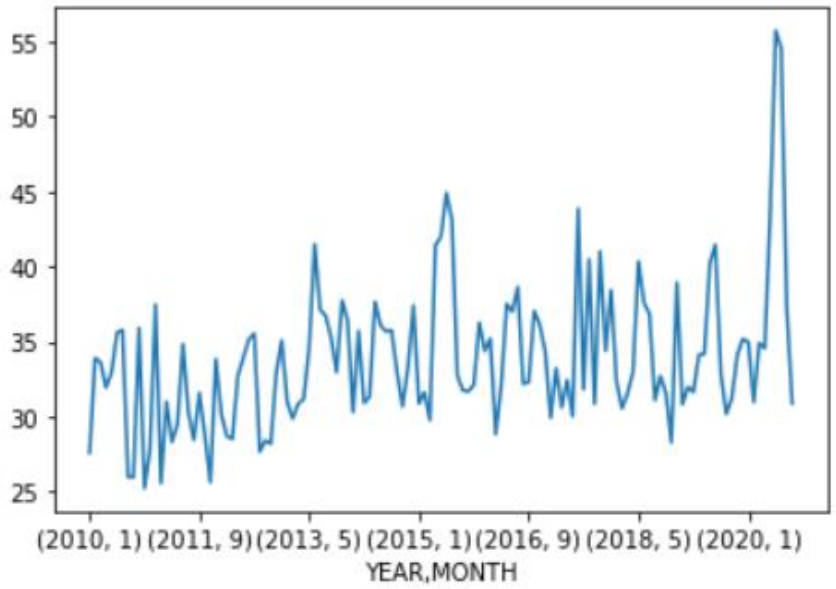
2. Wind speed in Shenzhen during the past 10 years

In this problem set, we will examine how wind speed changes in Shenzhen during the past `10` years, we will take a look at the hourly weather data measured at the BaoAn International Airport. The data set is from NOAA Integrated Surface Dataset. Download the file 2281305.zip, where the number `2281305` is the site ID. Extract the zip file, you should see a file named `2281305.csv`. Save the `.csv` file to your `working directory`.

Read page `8-9` of the comprehensive user guide for the detailed format of the wind data. Explain how you filter the data in your report.

[10 points] Plot monthly averaged wind speed as a function of the observation time. Is there a trend in monthly averaged wind speed within the past `10` years?

```
In [31]: df1.groupby(["YEAR","MONTH"]).mean()['WS'].plot()
Out[31]: <AxesSubplot:xlabel='YEAR,MONTH'>
```



3. Explore a data set

Browse the CASEarth, NOAA Land-Based Datasets and Products, or Advanced Global Atmospheric Gases Experiment (AGAGE) website. Search and download a data set you are interested in. You are also welcome to use data from your group in this problem set. But the data set should be in `csv`, `XLS`, or `XLSX` format, and have temporal information.

3.1 [5 points] Load the `csv`, `XLS`, or `XLSX` file, and clean possible data points with missing values or bad quality.

3.2 [5 points] Plot the time series of a certain variable.

3.3 [5 points] Conduct at least `5` simple statistical checks with the variable, and report your findings.

At first, I don't know how to download data. Then my classmate Deng Weihao told me.

Then I downloadZ2909.csv

```
#PS2_3.1
df=pd.read_csv('Z2909.csv')
df1=df[df['Event']!=-999]
df1
```

	#	Date Time	GMT+08:00	Event	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8
10	11	2016/7/24	����2h25��24��	1	NaN	NaN	NaN	NaN	NaN
11	12	2016/7/24	����2h30��03��	2	NaN	NaN	NaN	NaN	NaN
12	13	2016/7/24	����2h36��09��	3	NaN	NaN	NaN	NaN	NaN
14	15	2016/7/24	����2h44��15��	4	NaN	NaN	NaN	NaN	NaN
16	17	2016/7/24	����2h56��10��	5	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...
32401	32402	2018/10/5	����12h40��01��	8297	NaN	NaN	NaN	NaN	NaN
32402	32403	2018/10/5	����12h46��57��	8298	NaN	NaN	NaN	NaN	NaN
32403	32404	2018/10/5	����1h01��10��	8299	NaN	NaN	NaN	NaN	NaN
32408	32409	2018/10/5	����4h35��18��	8300	NaN	NaN	NaN	NaN	NaN
32451	32452	2018/10/6	����11h01��36��	8301	NaN	NaN	NaN	NaN	NaN

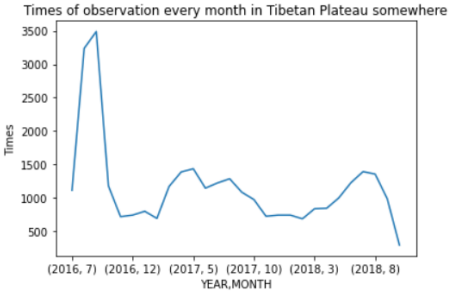
8301 rows × 9 columns

```
#PS2_3.2
# To split the column 'Date Time'
df['YEAR'] = pd.to_datetime(df['Date Time']).dt.year
df['MONTH'] = pd.to_datetime(df['Date Time']).dt.month

#To groupby
S1=df.groupby(["YEAR","MONTH"]).count()['Event']

#To plot
import matplotlib.pyplot as plt
plt.title("Times of observation every month in Tibetan Plateau somewhere")
plt.ylabel("Times")
S1.plot()
```

```
<AxesSubplot:title={'center':'Times of observation every month in Tibetan Plateau somewhere'}, xlabel='YEAR,MONTH', ylabel='Times'>
```



In [47]:
```
#PS2_3.3
#The maximum number of times of precipitation observed in this site every month
df1.groupby(["YEAR","MONTH"]).count()['Event'].max()
```

Out[47]: 758

In [48]:
```
#PS2_3.3
#The minimum number of times of precipitation observed in this site every month if there is at least a rain in this month
df1.groupby(["YEAR","MONTH"]).count()['Event'].min()
```

Out[48]: 6

In [66]:
```
#PS2_3.3
#The sum of the number of times of precipitation observed in this site during this time
df1.groupby(["YEAR","MONTH"]).count()['Event'].sum()
```

Out[66]: 8301

In [67]:
```
#PS2_3.3
#The average number of times of precipitation observed in this site every month if there is at least a rain in this month
df1.groupby(["YEAR","MONTH"]).count()['Event'].mean()
```

Out[67]: 345.875

In [69]:
```
#PS2_3.3
#The variance of the number of times of precipitation observed in this site every month if there is at least a rain in this month
df1.groupby(["YEAR","MONTH"]).count()['Event'].var()
```

Out[69]: 56733.85326086957