In [9]:

```
##################Question 1
import pandas as pd
fileName = 'E:\桌面\earthquakes-2022-10-18_18-49-57_+0800.tsv'
tsv_file=pd.read_csv(fileName , sep='\t', header=0)
data=tsv_file[["Location Name","Deaths","Mag"]]
data1=data["Location Name"].str.split(':',expand=True)
data.iloc[:,0]=data1[0]
coutry=[]
death=[]
for LocationName in data.iloc[:,0]:
    if LocationName in coutry:
        1
    else :
        coutry.append(LocationName)
for name in coutry:
   death.append(data.loc[data['Location Name']==name].sum()["Deaths"])
sort=sorted(death,reverse=True)
for i in range(0,20):
    print(coutry[death.index(sort[i])],sort[i])
```

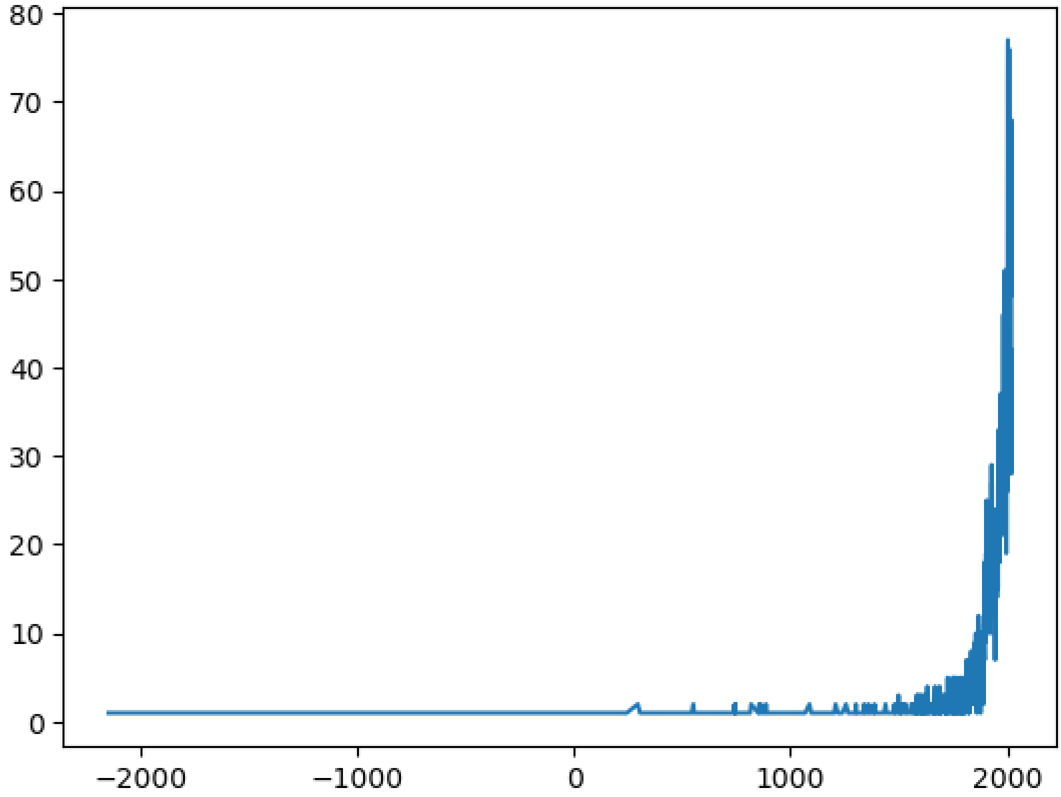
```
CHINA 2075019.0
TURKEY 1094479.0
IRAN 995403.0
ITALY 498477.0
SYRIA 369224.0
HAITI 323474.0
AZERBAIJAN 317219.0
JAPAN 277142.0
ARMENIA 191890.0
ISRAEL 160120.0
PAKISTAN 145080.0
ECUADOR 135479.0
IRAQ 120200.0
TURKMENISTAN 117412.0
PERU 101511.0
PORTUGAL 83506.0
GREECE 79278.0
CHILE 64269.0
INDIA 61940.0
TAIWAN 57134.0
```

In [39]:

```
data=tsv_file[["Location Name","Deaths","Mag","Year","Mo","Dy"]]
data=data.loc[(data["Mag"].astype(float)>3)]
data=data.loc[(data["Year"].astype(float)>-10000)]
list = data['Year'].value_counts()
list=list.sort_index()
list.plot()
```

Out[39]:

```
<AxesSubplot:>
```



In [62]:

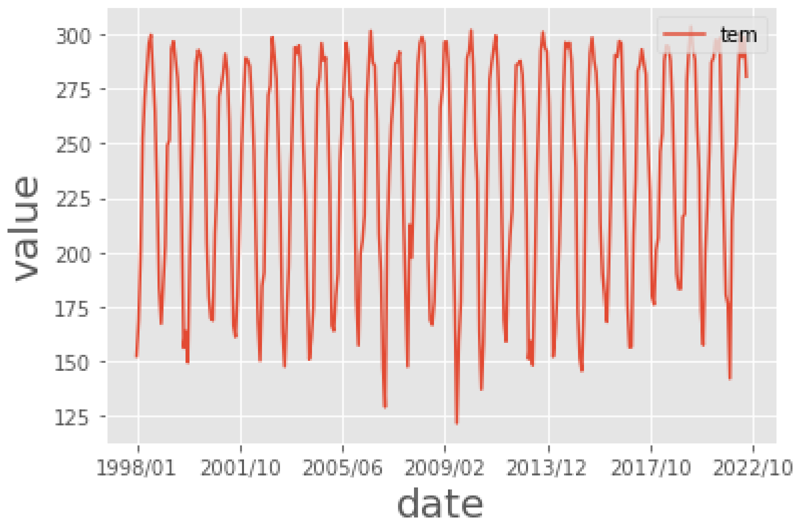
```python
def  CountEq_LargestEq(name):
    data2=data3.loc[(data3["County"]==name)]
    print("total count:",len(data2))
    print(data2.loc[(data2["Mag"]==data2["Mag"].max())])
data1=tsv_file[["Location Name","Mag"]]
data3=pd.concat([data1,data1["Location Name"].str.split(':',expand=True).iloc[:,0]],axis=1)
data3.columns=["Location Name","Mag","County"]
CountEq_LargestEq("CHINA")
```

```
total count: 616
                  Location Name  Mag County
977  CHINA:  SHANDONG PROVINCE  8.5  CHINA
```

In [3]:

```
########Question 2
import pandas as pd
import matplotlib.pyplot as plt
data=pd.read_csv(r”C:\Users\wangy\Desktop\Baoan_Weather_1998_2022.csv”)
data1=data[[”DATE”,”TMP”]]
data2=data1[”DATE”].str.split(’T’,expand=True)
data2=data2.iloc[:,0].str.split(’-’,expand=True)
data2=data2.iloc[:,0]+data2.iloc[:,1]
data3=data1[’TMP’].str.split(’,’,expand=True)
data4=pd.concat([data2,data3.iloc[:,0]],axis=1)
data4.columns=[’date’,’tem’]
data4=data4.loc[ (data4[’tem’].astype(int) <9999)]
data4=data4.astype(int)
dat=0
datelist=[]
templist=[]
for x in data4[’date’]:
    if dat!=x:
        dat=x
        datelist.append(dat)
        #print(data4.loc[(data4[’date’]==dat)][’tem’])
        a=data4.loc[(data4[’date’]==dat)].copy()
        templist.append(a[’tem’].mean())


date =[str(i) for i in datelist]
from datetime import datetime
import matplotlib.dates as mdates
from matplotlib.pylab import style
from PyQt5.QtGui import *
style.use(’ggplot’)
f, ax = plt.subplots()
plt.plot(templist,label=’tem’)
plt.xlabel(’date’,fontsize=20)
plt.ylabel(’value’,fontsize=20)
dates = [’0’,’1998/01’, ’2001/10’, ’2005/06’, ’2009/02’,’2013/12’,’2017/10’,’2022/10’,]
ax.set_xticklabels(dates)

plt.legend(loc=’upper right’)
plt.show()
```

```
C:\Users\wangy\AppData\Local\Temp\ipykernel_596\3454129979.py:3: DtypeWarning: Column
s (4,8,9,10,11,14,15,24,25,27,29,31,34,37,38,40,41,45,49,50) have mixed types. Specif
y dtype option on import or set low_memory=False.
  data=pd.read_csv(r”C:\Users\wangy\Desktop\Baoan_Weather_1998_2022.csv”)
C:\Users\wangy\AppData\Local\Temp\ipykernel_596\3454129979.py:36: UserWarning: FixedF
ormatter should only be used together with FixedLocator
  ax.set_xticklabels(dates)
```



In [4]:

```
####################Question 3
##################3-1
df=pd.read_csv(r″C:\Users\wangy\Desktop\ibtracs.ALL.list.v04r00.csv″)
df1=df[[″SID″,″NAME″,″WMO_WIND″]]
df1=df1.iloc[1:,:]
df1=df1.loc[(df1[’WMO_WIND’]!=’ ’)]
# df1=df1.loc[(df1[’WMO_WIND’].astype(int)>120)]
max10=pd.to_numeric(df1.get(’WMO_WIND’),errors=’coerce’).nlargest(10,keep=’all’)
x=0
for i in max10:
    if x!=i:
        x=i
        print(df1.loc[(df1[’WMO_WIND’].astype(int)==i)])
```

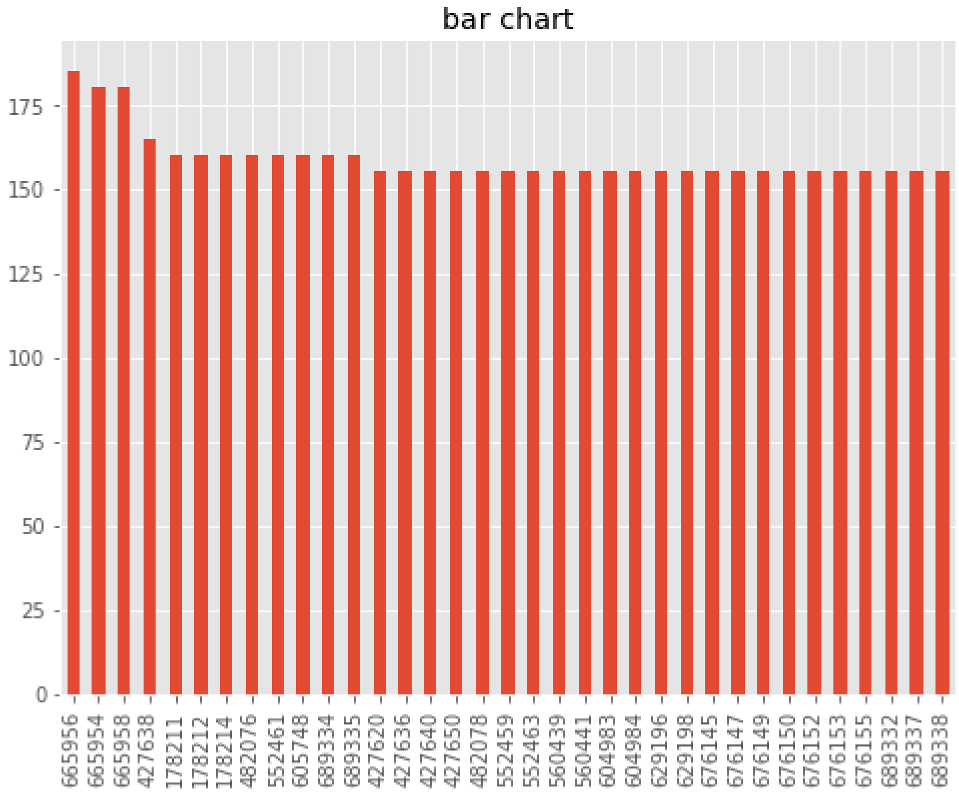
```
C:\Users\wangy\AppData\Local\Temp\ipykernel_596\12591636.py:3: DtypeWarning: Columns
(1,2,8,9,14,19,20,161,162) have mixed types. Specify dtype option on import or set lo
w_memory=False.
  df=pd.read_csv(r″C:\Users\wangy\Desktop\ibtracs.ALL.list.v04r00.csv″)

                  SID      NAME WMO_WIND
665956  2015293N13266  PATRICIA      185
                  SID      NAME WMO_WIND
665954  2015293N13266  PATRICIA      180
665958  2015293N13266  PATRICIA      180
                  SID   NAME WMO_WIND
427638  1980214N11330  ALLEN      165
                  SID       NAME WMO_WIND
178211  1935241N23291  NOT_NAMED      160
178212  1935241N23291  NOT_NAMED      160
178214  1935241N23291  NOT_NAMED      160
482076  1988253N12306    GILBERT      160
552461  1997253N12255      LINDA      160
605748  2005289N18282      WILMA      160
689334  2019236N10314     DORIAN      160
689335  2019236N10314     DORIAN      160
```

In [5]:

```
########3-2
max20=pd.to_numeric(df1.get('WMO_WIND'),errors='coerce').nlargest(20,keep='all')
x=0
data20=df1.loc[(df1['WMO_WIND'].astype(int)>10000)]
for i in max20:
    if x!=i:
        x=i
        data20=pd.concat([data20,df1.loc[(df1['WMO_WIND'].astype(int)==i)]])
data20=data20.WMO_WIND.astype(int)
data20.plot.bar(figsize = (8,6),x='SID',y='WMO_WIND',title='bar chart')
```

Out[5]:

```
<AxesSubplot:title={'center':'bar chart'}>
```



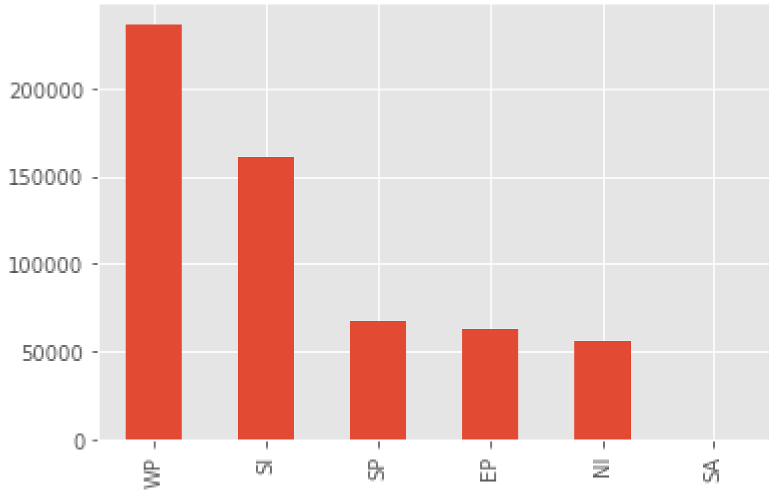
In [6]:

```
############3-3
df3=df["BASIN"]
df3=df3.iloc[1:]
counts=df3.value_counts()
print(counts)
counts.plot.bar()
```

```
WP    236576
SI    160668
SP     67119
EP     62412
NI     55402
SA       119
Name: BASIN, dtype: int64
```

Out[6]:

```
<AxesSubplot:>
```

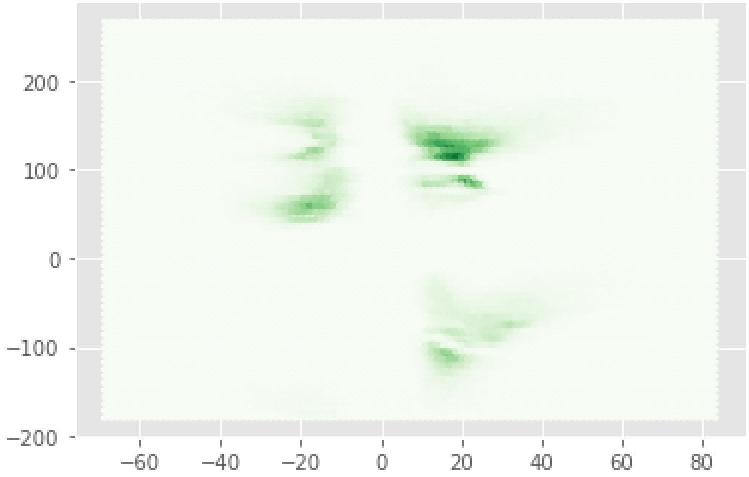


In [7]:

```
###################3-4
df4=df[[″LAT″,″LON″]]
df4=df4.iloc[1:]
plt.hexbin(df4[’LAT’].astype(float), df4[’LON’].astype(float), gridsize = 100, cmap =’Greens’)
```

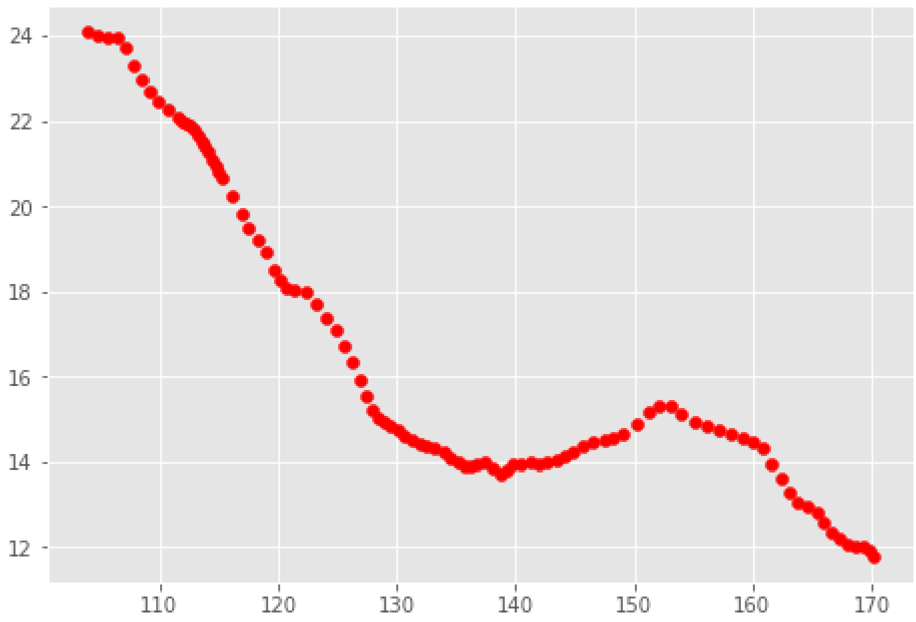
Out[7]:

```
<matplotlib.collections.PolyCollection at 0x221853f73a0>
```



In [8]:

```
# 3.5
df_name = df.groupby(["NAME"])
for name,group in df_name:
    if(name == "MANGKHUT"):
        df5 = group
df5 = df5.loc[df5["ISO_TIME"].astype(str).str.contains("2018-")]
lon = df5["LON"]
lat = df5["LAT"]
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.scatter(lon,lat,color = 'r')
plt.show()
```



In [9]:

```
####################3-6
df6=df[["SEASON","ISO_TIME","BASIN","SUBBASIN"]]
df6=df6.iloc[1:,:]
df61=df6.loc[(df6['BASIN']=='WP')]
df62=df6.loc[(df6['BASIN']=='EP')]
df6=pd.concat([df61,df62])
df6=df6.loc[(df6['SEASON'].astype(int)>1970)]
print(df6)
```

```
       SEASON             ISO_TIME BASIN SUBBASIN
357394   1971  1971-01-08 00:00:00    WP       MM
357395   1971  1971-01-08 03:00:00    WP       MM
357396   1971  1971-01-08 06:00:00    WP       MM
357397   1971  1971-01-08 09:00:00    WP       MM
357398   1971  1971-01-08 12:00:00    WP       MM
...       ...                  ...   ...      ...
707083   2022  2022-10-10 06:00:00    EP       MM
707084   2022  2022-10-10 09:00:00    EP       MM
707085   2022  2022-10-10 12:00:00    EP       MM
707086   2022  2022-10-10 15:00:00    EP       MM
707087   2022  2022-10-10 18:00:00    EP       MM

[172797 rows x 4 columns]
```

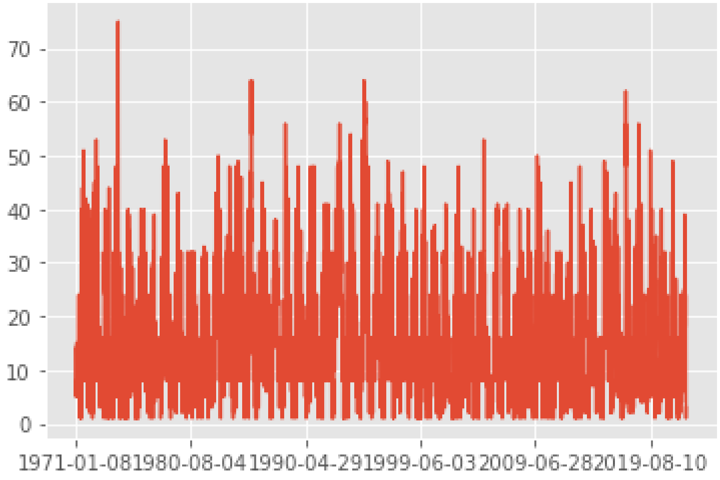
In [10]:

```
#####################3-7
df7=df6.ISO_TIME
df7=df7.str.split(' ',expand=True)
df6.ISO_TIME=df7.iloc[:,0]
counts=df6['ISO_TIME'].value_counts()
counts=counts.sort_index()
print(counts)
counts.plot()
```

```
1971-01-08    14
1971-01-09     8
1971-01-10     8
1971-01-11     8
1971-01-12     8
              ..
2022-10-04     9
2022-10-05     7
2022-10-09     1
2022-10-10     7
2022-10-12     3
Name: ISO_TIME, Length: 10619, dtype: int64
```

Out[10]:

```
<AxesSubplot:>
```



In [11]:

```
#####################3-8
def countdays(year,month,day):
    days=0
    dic={'1':31,'2':28,'3':31,'4':30,'5':31,'6':30,'7':31,'8':31,'9':30,'10':31,'11':30,'12':31}
    if year%4 == 0 and year%100 ==0 or year%400 == 0:
        dic['2']=29
    if int(month)>1:
        for obj in dic.keys():
            if month==int(obj):
                for i in range(1,int(obj)):
                    days+=dic[str(i)]
        days+=day
    else:
        days=day
    return days

df8=df6.ISO_TIME.str.split('-',expand=True)
df8.columns=["year","month","day"]
list=[]
for j in range(0,len(df8)):
    list.append(countdays(int(df8.iloc[j,0]),int(df8.iloc[j,1]),int(df8.iloc[j,2])))
result=pd.value_counts(list)
result=result.sort_index()
print(result)
result=result.tolist()
#the frenquency in 366 days
```

```
1       83
2       72
3       74
4       93
5      105
      ...
362    147
363    130
364     98
365     85
366      8
Length: 366, dtype: int64
```

In [12]:

```
####################3-9
mean = sum(result)/len(result)
for i in range(0,366):
    print("day",i,":",result[i]-mean)
```

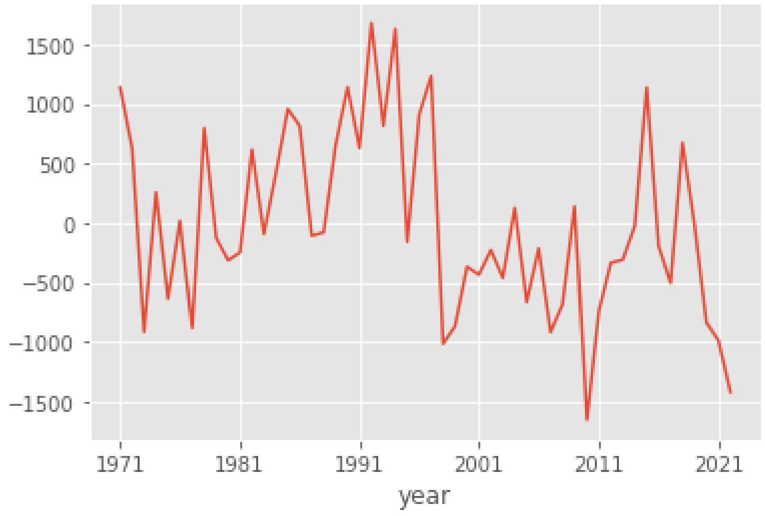
```
day 0 : -389.12295081967216
day 1 : -400.12295081967216
day 2 : -398.12295081967216
day 3 : -379.12295081967216
day 4 : -367.12295081967216
day 5 : -351.12295081967216
day 6 : -359.12295081967216
day 7 : -327.12295081967216
day 8 : -334.12295081967216
day 9 : -335.12295081967216
day 10 : -325.12295081967216
day 11 : -316.12295081967216
day 12 : -305.12295081967216
day 13 : -337.12295081967216
day 14 : -366.12295081967216
day 15 : -391.12295081967216
day 16 : -394.12295081967216
day 17 : -400.12295081967216
day 18 : -402.12295081967216
day 19 : -410.12295081967216
```

In [13]:

```
####################3-10
df9=df8.copy()
df9["times"]=1
df10=df9[["year","times"]].groupby("year").count()
mean10=df10["times"].mean()
df10["anomaly"]=df10["times"]-mean10
df10["anomaly"].plot()
```

Out[13]:

```
<AxesSubplot:xlabel='year'>
```



In [14]:

```
#Question 4
#4-1
rain=pd.read_csv(r"C:\Users\wangy\Desktop\dailyrain.csv")
rain.dropna()
```

Out[14]:

	date	G1183	G1193	G3515	G3528	G3529	G3531	G3532	G3550	G3557	...	G368
0	01/01/2017	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.
1	02/01/2017	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.
2	03/01/2017	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.
3	04/01/2017	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.
4	05/01/2017	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.
...	...	...	...	...	...	...	...	...	...	...	...	.
1090	27/12/2019	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.
1091	28/12/2019	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.
1092	29/12/2019	1.2	1.2	1.8	1.9	1.2	2.5	1.6	1.1	1.4	...	1.
1093	30/12/2019	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	...	0.
1094	31/12/2019	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.

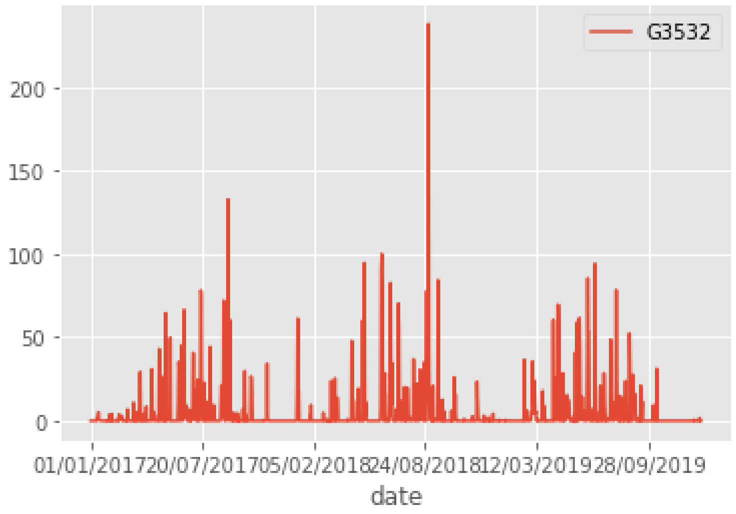
1095 rows × 22 columns

In [15]:

```
#4-2
rain1=rain[["date","G3532"]]#the rainfall of station G3532 from 2017-01-01 to 2019-10-30
rain1.plot(x='date', y='G3532')
```

Out[15]:

```
<AxesSubplot:xlabel='date'>
```



In [17]:

```
#4-3
print(rain.corr())
print(rain.cov())
```

```
G3528     190.726504  200.550145  167.976248  200.127693  192.549658
G3529     199.806711  217.410104  173.968419  210.954724  188.644084
G3531     180.993764  180.006099  161.194820  183.941009  173.539257
G3532     191.020298  230.552353  191.135550  227.919330  195.241865
G3550     178.569453  204.661575  176.236798  205.732546  173.536044
G3557     198.067205  208.886041  195.002861  210.439953  188.452564
G3583     207.075738  247.395221  207.896636  248.000556  208.643372
G3682     159.627999  158.411427  138.768955  163.345697  151.087221
G3686     182.207865  206.026866  168.706876  201.080515  179.434649
G3693     177.654143  183.548404  160.330245  183.755976  177.551064
G3701     160.012671  170.858345  141.880680  170.378728  163.738633
G3722     184.278815  196.428323  166.798383  196.406116  186.345256
G3727     184.816146  181.725118  155.355755  184.283097  176.579619
G3728     206.207751  206.843510  176.438735  203.346965  185.966485
G3756     206.843510  247.056049  197.271272  240.909127  202.545610
G3781     176.438735  197.271272  190.099122  198.757993  171.560230
G3785     203.346965  240.909127  198.757993  245.070832  201.431682
average   185.966485  202.545610  171.560230  201.431682  181.179140

[21 rows x 21 columns]
```

In [18]:

```
def status(x) :
    return pd.Series([x.count(),x.quantile(.25),x.median(),
                      x.quantile(.75),x.mean(),x.max(),x.idxmax(),x.mad(),x.var(),
                      x.std(),x.skew(),x.kurt()],index=['总数','25%分位数',
                   '中位数','75%分位数','均值','最大值','最大值位数','平均绝对偏差','方差','标准差'

status(rain["average"])
```

Out[18]:

```
总数          1095.000000
25%分位数        0.000000
中位数           0.010000
75%分位数        2.270000
均值            4.615799
最大值         189.550000
最大值位数       606.000000
平均绝对偏差        6.821833
方差          181.179140
标准差          13.460280
偏度            5.744239
峰度           48.267327
dtype: float64
```

In [ ]: