```
①  from os import list dir
②  from os.path import is file
③  from os.path import join
④  from shapely.geometry import Polygon
⑤  from geopy.distance import great_circle
⑥  from geopandas import GeoDataFrame
⑦  from descartes import PolygonPatch
⑧  import Pandas as pd
⑨  import numpy as np
⑩  import matplotlib as plt
⑪  import matplotlib.font_manager as fm
⑫  import date time


⑬  myPath = "/users/local/Sem-3/capstone/
⑭  allDir = {}
⑮  i = 0
⑯  files = [f for f in list dir(my Path) \
⑰          if is file (join (my path, f))]


⑱  for file in files :
⑲      df = pd.read_csv (file)
⑳      Print (df)
㉑      allDic [i] = df
```

```
22  x = [row.split(" ")[0] for row in df]
23  y = [row.split(" ")[1] for row in df]

24  fig = plot.figure()
25  ax = fig.add_subplot(11)
26  ax.set_title("plot of pollution data")
27  ax.set_xlabel("x axis")
28  ax.set_ylabel("y axis")
29  ax.plot(x, y, c='r', label="the data pollution")
30  legend = ax.legend()
31  plt.show()

32  liAll = []
33  for each in allDir:
34      for record in each:
35          liAll.append(record)
36  print(liAll)

37  fig = plt.figure(figsize=(x 120, 120))
38  ax = fig.add_subplot(11)
39  ax.set_axis_bgcolor(background colour = "black")
40  print(fig)
```

```
(41)   for patch in get-patches (locations, time series)
(42)       ax.add-patch (patch)
(43)   Points-scatter = ax. scatter (x = points ["x"],
(44)       y = points ['y'], c = 'm', alpha = 0.4, s=100)
(45)   ax.set =title ("plots of Pollution data 2")
(46)   ax. xlim (500)
(47)   ax. ylim (500)
(48)   ax. xticks ([])
(49)   ax. yticks ([])
(50)   for i, row in most. interows():
(51)       plt. annotate (row ['city'],
(52)           xy = (row ['x'], row['y']),
(53)           xytext = (row ['x'] = 35800, row['y'] =10
(54)           bbox = dt (boxstyle = 'round',
(55)           color = 'x', fc = 'w', alpha =0.8),
(56)           xycord = 'data')
(57)   plt. show ()
(58)   for each in liAll:
(59)       xAxis = np. array (diuAll [all key])
(60)       yaxis = np. array (duAll [all value])
(61)       plt. plot (x, y)
(62)   plt. show ().
```

```
(63)  dates = np.loadertxt (page = sum Allate; upack = True;
(64)         converter = {0 : nadates. stapdate2num ("%y-%y")}

(65)  plt.plot_date ( x = days, y = impressions, fmt="i-")
(66)  plt.title ("plots of pollution data 2")
(67)  plt.ylable ("pots lebels")
(68)  plt.grid (True)
(69)  plt.show ()
(70)  Series = Series.from_cv ("path to csv", header=0)
(71)  group = seri.groupby (Time.groups ('A'))
(72)  years = Dataframe y
(73)  for name, group in groups:
(74)         years[name. year] = group.values
(75)   years.plot (subplots=True, legent=False)
(76)  pyplot.show()
(77)  Series.hist ()
(78)  pyplot.show()
(79)  Series.plot (kind="kde")
(80)  pyplot.show()
(81)  groups = Series.groupby (Time.groups ('A'))
(82)  years = Dataframe()
```

```python
83   for date in groups:
84       years[name.year] = group.values
85   years.boxplot()
86   pyplot.show()
87   groups = series.groupby(Time grouper ("A"))
88   years = Dataframe()
89   for name, group in groups:
90       years[name.year] = group.values
91   years = years.T
92   pyplot.matshow(years, interpolation=none, \
93                  aspect="auto")
94   pyplot.show()
95   one-year = series(1990)
96   groups = one-year.groupby(Time grouper [N])
97   months = concat([Dataframe(x[1].values) \
98                    for x in groups], axis=1)
99   months = Dataframe(months)
100  months.columns = range(1, 13)
101  pyplot.matshow(months, interpolation=none \
102                 aspect ration = "auto")
103  pyplot.show()
```

```
(104)   locDic = {}
(105)   for all in allDic:
(106)       for one in all:
(107)           if location is in locDic:
(108)               locDic [one[1)] = loc Dic [one[1]]
(109)                               + one[0]
(110)           else:
(111)               locDic [one[1]) = one[0]
(112)   Print ( locDic)
(113)   max = 0
(114)   min = 0
(115)   for a in locDic:
(116)       if a > max:
(117)           max = a
(118)       if a < min:
(119)           min = a

(120)   Print ("maximum of all locations : " + max)
(121)   Print (" minimum of all locations : " + min)
```