1. 至少实现一个数据分析类，以提供数据的读取及基本的时间（如某区域某类型污染物随时间的变化）和空间分析（某时间点或时间段北京空气质量的空间分布态势）方法。

```python
class Data():  # 基类Data
    _instance = None
    def __init__(self, folder):
        self.df = Data.load(self, folder)
        self.variables = list(self.df)
        self.groups = self.df.groupby(['station', 'year']) # 按地区和年分组
        self.mean_year =  self.groups.mean()  # 按年求均值

    def load(self, folder):
        datas = []
        for path in Path(folder).glob('*.csv'):  # 读取文件夹中所有csv文件
            with open(path,'rt') as f:
                data = pd.read_csv(f)
                try:  # 异常捕捉
                    self.check(data, path.name)
                except NotNumError as error:
                    print(error.message)
                    data = data.fillna(method='bfill')  # 填充缺失值
                    self.check(data, path.name)
            datas.append(data)
        return pd.concat(datas)

    def check(self, data, file):
        col_names = list(data)
        for header in col_names:  # 检查缺失值
            for value in data[header]:
                if True == pd.isna(value):
                    if header in (col_names[:5] + col_names[-1:]):
                        exec(f'raise NotNumError(file, {header} = "{header}")')
                    elif header in col_names[5:11]:
                        raise NotNumError(file, pollutant = 'pollutant')
                    elif header in col_names[11:17]:
                        raise NotNumError(file, meteorological = 'meteorological')
```

```python
class Analysis(Data):
    def year(self, feature, station):
        print(station, feature)
        print(self.df.query(f'station == "{station}"').groupby('year')[feature].describe())

    def month(self, feature, station, year):
        d = self.groups.get_group((station, year))
        print(station, feature, year)
        sta = d.groupby('month')[feature].describe()
        print(sta)

    def space(self, feature, year):
        print(feature)
        d = self.mean_year.query(f'year == {year}')[feature]
        print(d)

    def corr(self):  # person相关系数
        return self.df.corr().iloc[lambda i:[11,12,13,15], 5:11]
```

东四 PM2.5 的年和月水平上的描述性统计特征：

Dongsi PM2.5

year	count	mean	std	min	25%	50%	75%	max
2013	7344.0	86.764706	76.593310	3.0	28.0	66.0	124.00	520.0
2014	8760.0	87.718858	85.165654	3.0	23.0	65.0	125.00	737.0
2015	8760.0	87.210868	91.984616	3.0	22.0	58.0	117.25	685.0
2016	8784.0	79.780168	82.126774	3.0	20.0	54.0	110.00	695.0
2017	1416.0	101.953390	122.599780	3.0	15.0	52.5	147.00	681.0

Dongsi PM2.5 2016

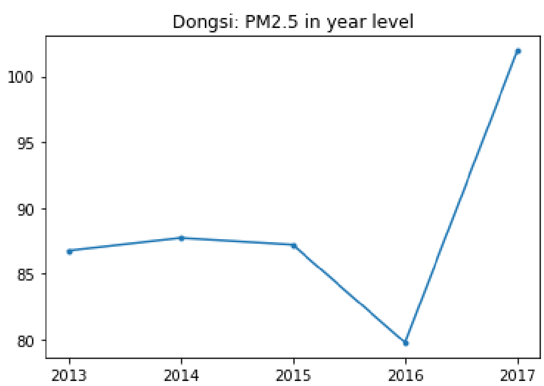
month	count	mean	std	min	25%	50%	75%	max
1	744.0	69.350806	81.363014	3.0	12.00	32.0	111.00	535.0
2	696.0	43.165230	75.435045	3.0	9.00	17.0	48.00	695.0
3	744.0	98.526882	102.074693	3.0	13.00	59.5	143.25	401.0
4	720.0	76.986111	62.007804	3.0	29.75	64.5	99.00	295.0
5	744.0	63.451613	54.103513	3.0	31.00	51.0	78.00	408.0
6	720.0	67.573611	44.429721	3.0	35.00	56.0	97.00	225.0
7	744.0	74.022849	51.604910	3.0	26.75	71.0	113.00	274.0
8	744.0	51.002688	35.143811	3.0	21.00	44.5	75.00	187.0
9	720.0	60.865278	53.924662	3.0	19.00	42.0	90.00	262.0
10	744.0	91.973118	78.302167	3.0	29.75	62.0	137.25	342.0
11	720.0	108.279167	88.622332	3.0	30.00	87.5	165.00	382.0
12	744.0	149.627688	138.566453	3.0	21.00	122.5	228.25	558.0

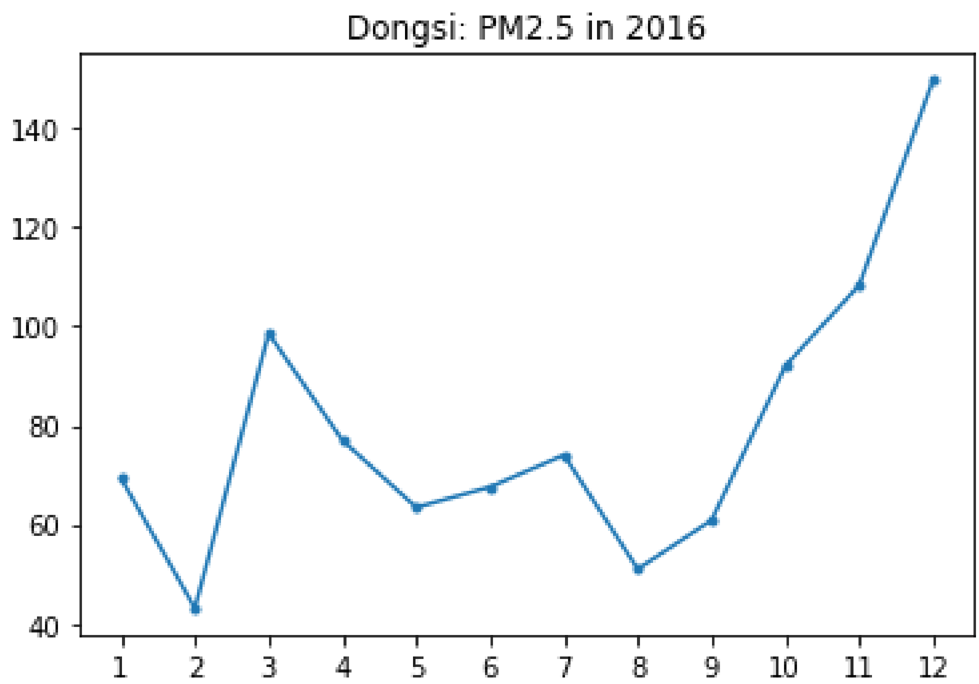
各地区 2016 年 PM2.5 均值

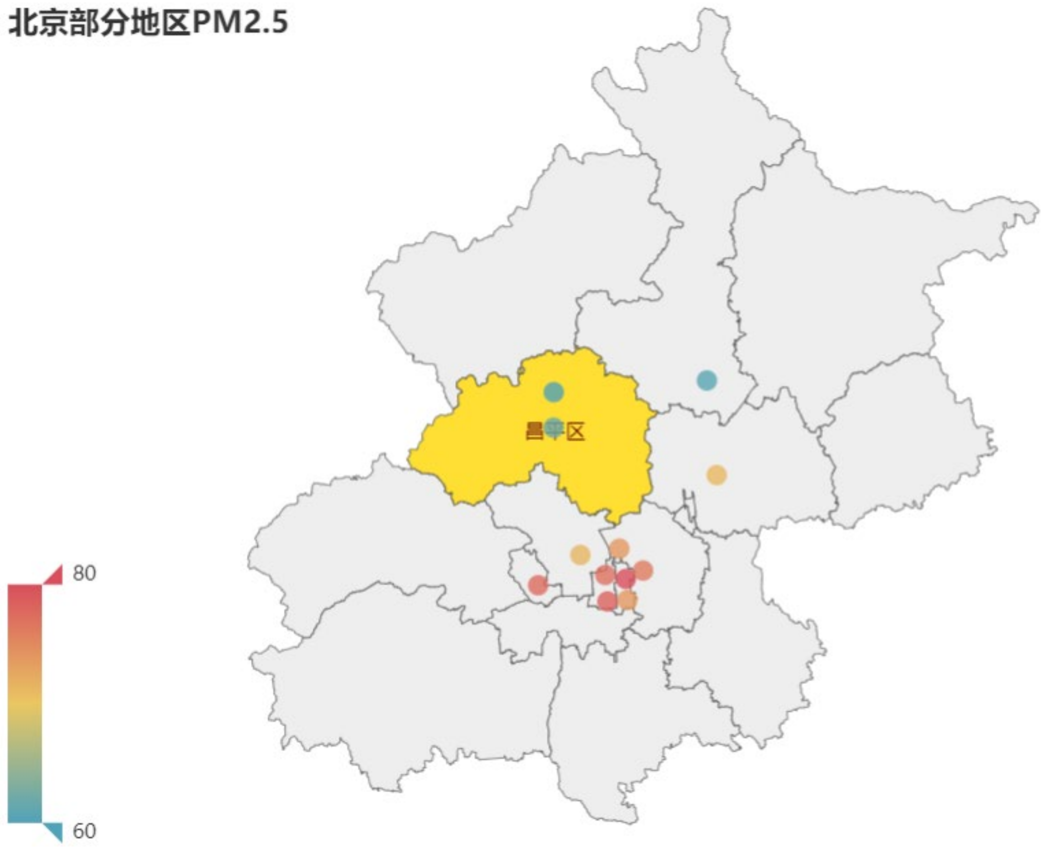
station	year	PM2.5
Aotizhongxin	2016	73.700137
Changping	2016	61.746357
Dingling	2016	60.247723
Dongsi	2016	79.780168
Guanyuan	2016	76.965164
Gucheng	2016	77.434199
Huairou	2016	60.895947
Nongzhanguan	2016	76.101434
Shunyi	2016	71.202527
Tiantan	2016	73.780852
Wanliu	2016	71.462204
Wanshouxigong	2016	78.207081

2. 至少实现一个数据可视化类，以提供上述时空分析结果的可视化，如以曲线、饼、地图等形式对结果进行呈现。

```python
class Visualization(Data):
    def plot_year(self, feature, station):
        d = self.mean_year.loc[station]
        plt.plot(d[feature], '.-')
        plt.xticks(d.index)
        plt.title(f'{station}: {feature} in year level')
        plt.show()

    def plot_month(self, feature, station, year):
        d = self.groups.get_group((station, year))
        d = d.groupby('month').mean()
        plt.plot(d[feature], '.-')
        plt.xticks(d.index)
        plt.title(f'{station}: {feature} in {year}')
        plt.show()

    def space(self, feature, year):
        d = self.mean_year.query(f'year == {year}')[feature]
        g = Geo().add_schema(maptype = '北京')
        location = [('Aotizhongxin', 116.406138, 39.990549),
                    ('Changping', 116.232154, 40.230159),
                    ('Dingling', 116.232569, 40.300468),
                    ('Dongsi', 116.423284, 39.93053),
                    ('Guanyuan', 116.36879, 39.93825),
                    ('Gucheng', 116.191305, 39.917885),
                    ('Huairou', 116.637667, 40.323608),
                    ('Nongzhanguan', 116.468772, 39.946988),
                    ('Shunyi', 116.663674, 40.136136),
                    ('Tiantan', 116.427392, 39.888497),
                    ('Wanliu', 116.30297, 39.977844),
                    ('Wanshouxigong', 116.37434, 39.885845)]
        i = 0
        for p in location:
            g.add_coordinate(p[0], p[1], p[2])  # 添加点
            g.add('', [(p[0], d.iloc[i])])  # 显示点
            i += 1
        g.set_series_opts(label_opts = opts.LabelOpts(is_show=False))
        g.set_global_opts(
            visualmap_opts=opts.VisualMapOpts(min_ = min(d), max_ = max(d)),
            title_opts=opts.TitleOpts(title = f'北京部分地区{feature}')
            )
        g.render(f'geo-{feature}.html')

    def corr(self):
        correlation = self.df.corr().iloc[lambda i:[11,12,13,15], 5:11]
        h = HeatMap().add_xaxis(list(correlation.index))
        h.add_yaxis("", list(correlation),
                    [(i, j, correlation.iloc[i,j]) for i in range(4) for j in range(6)])
        h.set_global_opts(
            visualmap_opts=opts.VisualMapOpts(min_ = -1, max_ = 1),
            title_opts=opts.TitleOpts(title = '污染物与气象相关系数')
            )
        h.render('correlation.html')
```







3. 如果数据中包含空值等异常值（可人工注入错误数据以测试程序），在进行数据分析以及可视化前需要检查数据。因此需要实现 NotNumError 类，继承 ValueError，并加入新属性 region, year，month，day,hour，pollutant，对数据进行检测，若取到的一列数据中包含空值等明显错误，则抛出该异常，并提供异常信息。在此基础上，利用 try except 捕获该异常，打印异常信息，并对应位置的数据进行适当的填充。

```python
class NotNumError(Exception):
    def __init__(self, file, No = '', station = '', year = '', month = '',
                 day = '', hour = '', pollutant = '', meteorological = ''):
        self.no = No
        self.station = station
        self.year = year
        self.month = month
        self.day = day
        self.hour = hour
        self.pollutant = pollutant
        self.message = f'\033[1;31mAttention:\033[0m In {file}, {No}{station}\
{year}{month}{day}{hour}{pollutant}{meteorological} has missing values!\n\
           And all missing values will be filled backward!'
```

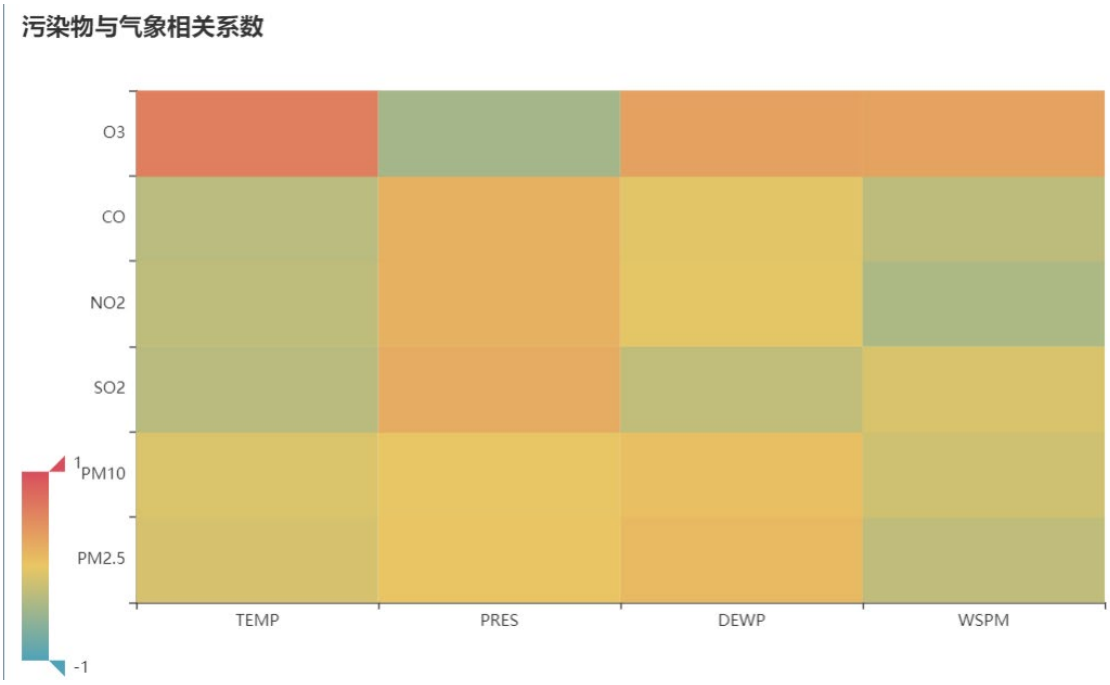
```python
    def load(self, folder):
        datas = []
        for path in Path(folder).glob('*.csv'):  # 读取文件夹中所有csv文件
            with open(path,'rt') as f:
                data = pd.read_csv(f)
                try:  # 异常捕捉
                    self.check(data, path.name)
                except NotNumError as error:
                    print(error.message)
                    data = data.fillna(method='bfill')  # 填充缺失值
                    self.check(data, path.name)
            datas.append(data)
        return pd.concat(datas)

    def check(self, data, file):
        col_names = list(data)
        for header in col_names:  # 检查缺失值
            for value in data[header]:
                if True == pd.isna(value):
                    if header in (col_names[:5] + col_names[-1:]):
                        exec(f'raise NotNumError(file, {header} = "{header}")')
                    elif header in col_names[5:11]:
                        raise NotNumError(file, pollutant = 'pollutant')
                    elif header in col_names[11:17]:
                        raise NotNumError(file, meteorological = 'meteorological')
```

```
Attention: In PRSA_Data_Aotizhongxin_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Changping_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Dingling_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Dongsi_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Guanyuan_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Gucheng_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Huairou_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Nongzhanguan_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Shunyi_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Tiantan_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Wanliu_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
Attention: In PRSA_Data_Wanshouxigong_20130301-20170228.csv, pollutant has missing values!
           And all missing values will be filled backward!
```

4.（附加）污染物含量与气象状态本身是否有相关性？请丰富数据分析类和数据可视化类，增加关于这些相关性探索的方法。

污染物含量与气象状态的相关系数矩阵：

	PM2.5	PM10	SO2	NO2	CO	O3
TEMP	-0.132215	-0.095126	-0.320713	-0.281421	-0.318546	0.594171
PRES	0.020550	-0.019203	0.221975	0.180178	0.183838	-0.446651
DEWP	0.112445	0.067958	-0.266656	-0.037482	-0.054677	0.312012
WSPM	-0.271813	-0.178076	-0.108240	-0.393261	-0.291939	0.296764



从热力图中可以看到，污染物含量与气象状态整体相关性不强，其中相关性最强的 O_3 与 TEMP 也只有 0.59。从理论上讲，气象状态是制约污染物在大气中稀释、扩散、迁移和转化的重要因素，本数据得到的结果可能是数据量较少、缺失值较多、选址较少等原因所致

污染物在水平方向上的扩散由风速决定，风速越大，污染物越容易扩散，风速小甚至静风，时，污染物难以扩散，容易形成污染物局地积累；污染物在垂直方向的扩散受到垂直方向上温度的分布状况控制，当地面空气温度高于高空中大气温度时，大气是不稳定的，在热力对流的作用下污染物向上扩散，地面污染物浓度降低，当高空中大气温度高于地面空气温度时，就形成了所谓的逆温现象，这时热力对流减弱甚至消失，大气状况变得稳定，污染物的垂直扩散受到抑制，地面污染物累积：降水(降雨、降雪)对空气污染物能起到清除和冲刷作用：在雨水作用下，大气中的一些污染气体能够溶解在水中，降低空气中污染气体的浓度，较大的雨雪对空气污染物粉尘颗粒也起着有效的清除作用。

5. （附加）思考不同区域时间变化的趋势及差异的管理意义。

从上面的空间可视化结果可以看到，PM2.5 浓度的分布具有明显的空间聚集性，以东城区、西城区逐渐向周围扩散。车辆、道路、扬尘和工业等的排放很可能对北京 PM2.5 的浓度影响较大。

空气污染物受季风、大气流动等因素影响，污染范围极易扩张。了解不同区

域时间变化的趋势及差异的管理意义，及时跟踪污染范围扩大的趋势，抓住污染源头，对了解污染源、主要污染原因、遏制污染扩散具有重要原因。