江西财经大學
JIANGXI UNIVERSITY OF FINANCE AND ECONOMICS

课程名称：Python语言与数据分析

课 程 报 告

项目名称 从天气角度分析我更应该在哪度过暑假
——以家乡甘肃金昌和现居地江西南昌为例

班　　级 金融202

学　　号 0204824

姓　　名 谢晓蕾

任课教师 肖　泉

开课学期： 2020 至 2021 学年第二学期

完成时间： 2020 年 7 月 4 日

《从天气角度分析我更应该在哪度过暑假
——以家乡甘肃金昌和现居地江西南昌为例》
数据分析报告

目　录

目录

1 概述

项目研究背景

本数据分析项目作者生源地位于甘肃金昌，现就读于江西财经大学，即现居地位于江西南昌。甘肃省位于中国西北地区，江西省位于中国东南地区，其二者包括但不限于地貌、水文、植被、气候在内的地理特征均表现出明显差别。

金昌市，位于甘肃省河西走廊东段，祁连山北麓，阿拉善台地南缘，介于东经101°04′35″—102°43′40″，北纬 37°47′10″—39°00′30″之间。金昌地势自西南向东北倾斜。地形以山地、平原为主，戈壁、绿洲、大漠东西展开，南北更替，相间排列。金昌属大陆性温带干旱气候，光照充足，气候干燥，全年多西北风，昼夜、四季温差较大，霜期长，春季多大风。境内气温北高南低，降水北少南多。由东北到西南，大体划分为五个气候区，即温和极干旱区和温凉干旱区；温寒干旱区;寒冷半干旱区和寒冷半湿润区；寒冷湿润区；高寒湿润区和高寒很湿润区。

南昌市地处江西中部偏北，赣江、抚河下游，鄱阳湖西南岸，位于东经 115°27'至116°35'、北纬 28°10'至 29°11'之间。全境以平原为主，占 35.8%，东南相对平坦，西北丘陵起伏，水网密布，湖泊众多。南昌市属于亚热带湿润季风气候，气候湿润温和，日照充足，一年中夏冬季长，春秋季短。南昌市地处北半球亚热带内，受东亚季风影响，形成了亚热带季风气候。冬季多偏北风，夏季多偏南风。市内热量丰富、雨水充沛，光照充足，且作物生长旺季雨热匹配较好，为农业生产提供了有利气象条件，素有鱼米之乡的美誉。

作者根据中国天气网给出的两地未来 1 天各小时与未来 14 天天气预报，基本可以研究出两座城市在 2021 年 7 月某一天和 7 月的天气差别并进行多种数据可视化，再根据个人偏好因素和体感舒适程度从而判断出 7 月间哪座城市相对更适合作者居住。

项目研究意义

浏览天气预报与研究天气情况对于人们购买雨伞还是遮阳帽、防潮用品还是加湿器及选择居住城市、居住环境等各方面均有重要意义，对于酒店业分店开设、企业生产、超市进货等商业活动也有一定参考价值，因此针对天气数据做数据分析报告具有较高的研究价值。

本项目研究意义不仅在于研究该两座城市在 7 月某天或者 7 月间的天气情况并做出宜居度分析，事实上本项目的数据分析代码设计出了一套通用的完整模板，内含多种计算公式和可视化设计，通过本套代码只需进行简单的操作（如改变城市代码）就可以对比研究中国天气网内其他多个城市的未来天气状况，其中涉及有空气污染度、风级风向、区间内各天气日数、温差等多种因素，具备初步得到合理结论的条件，对于人们在日常生产生活中的选择具有参考价值。

2 数据描述

数据来源与提取

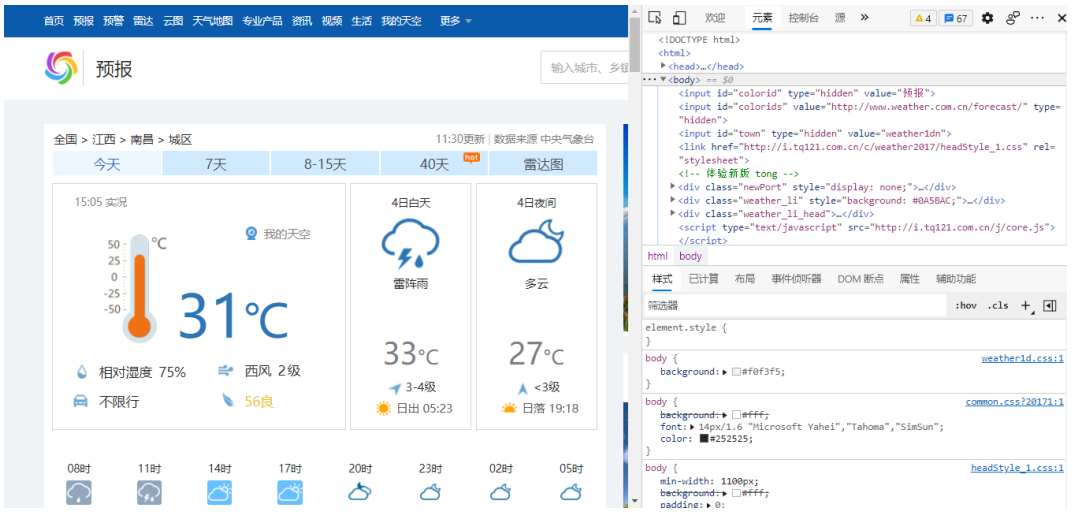
本数据分析报告内的气象数据来源于中国天气网（http://www.weather.com.cn/），所用到的其中与南昌相关的网址有：

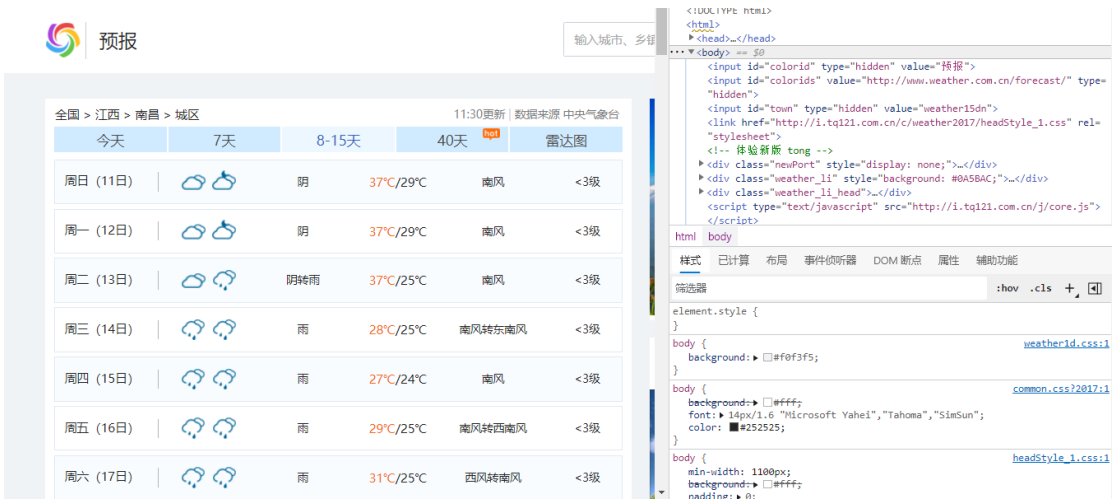
南昌 1 天天气预报（http://www.weather.com.cn/weather1d/101240101.shtml）

南昌 8-15 天天气预报（http://www.weather.com.cn/weather15d/101240101.shtml）（金昌同上）

首先，通过研究网址可知，在中国天气网爬取数据时只需要改变本网址中 weather 后的数字和城市代码即可，1d 代表 1 天数据，主要呈现该天内各时段的不同数据；15d 代表未来 8-15 天天气数据，可以反映未来天气变化趋势。

下图展示对中国天气网爬取数据时的一些探索。





打开需要提取的网页，右键检查提取特征标签，利用 requests 库获取网页源代码，利用

BeautifulSoup 库提取天气信息，爬取后保存为 CSV 文件，再用 matplotlib、numpy、pandas 库对数据进行可视化处理，最终获得温差变化曲线、空气质量图、风向雷达图等图像。

数据处理

通过分析标签可知，7 天数据信息在 div 中的“7d”，日期、天气和其他信息位于 ul 和 li，所以运用 BeautifulSoup 查找“7d”并提取出所有 ul 和 li，比如 ul = data.find(‘ul’)找到所有需要的标签。

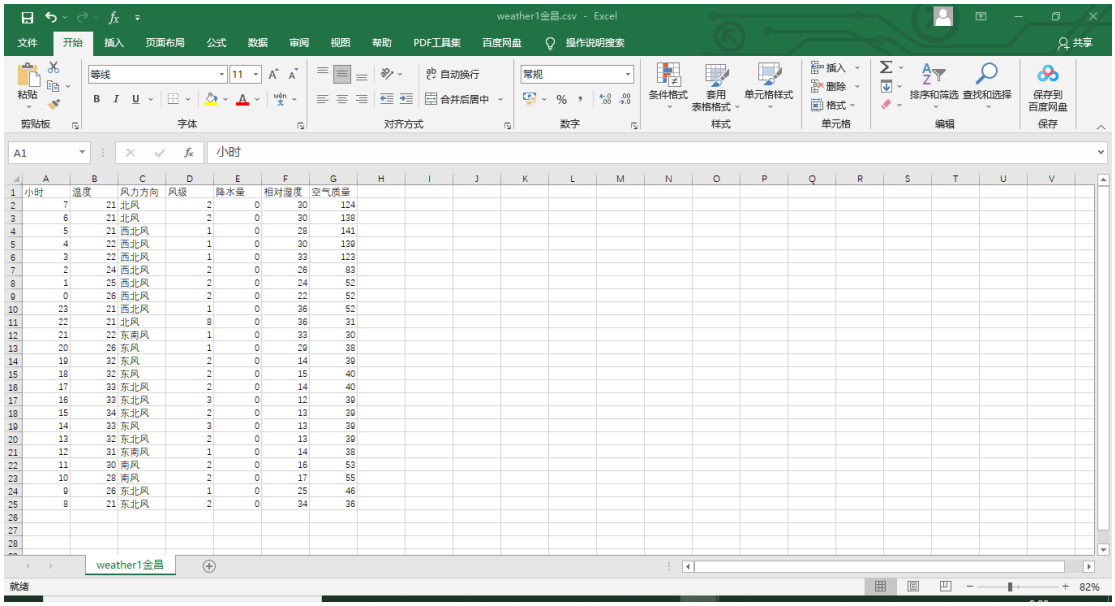
同时，检查数据格式是否正确，提前进行数据保存格式的处理，比如日期里的数字。此时可能也会出现部分数据提取不全，因此需要设计 if xxx is None else xxx 的结构避免提取过程出现问题。

将爬取的数据添加到列表中，运用 CSV 库将所需要的四组数据分别保存，在进行下一步前多次检查生成的 csv 文件，可以和网站中的数据做对比，确保进行可视化前的数据精准无误。

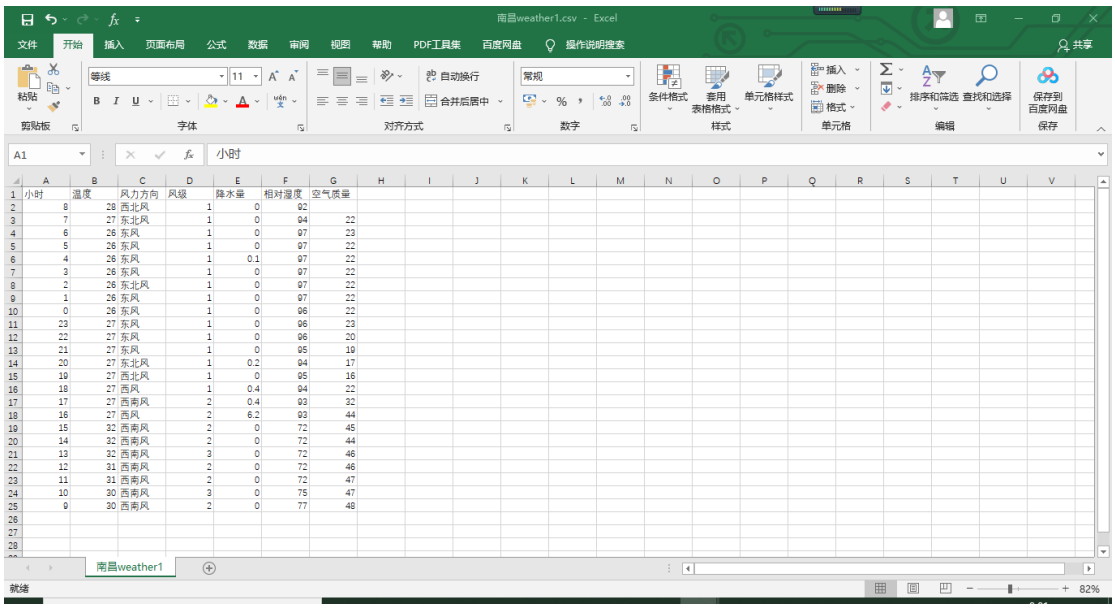
3 数据分析内容

分别爬取中国天气网里金昌和南昌的 1 天信息与 14 天信息，得到两个含有小时、温度、风力风向、风级、降水量、相对湿度、空气质量的 1 天信息文件，两个含有 7 月 5 日-7 月 18 日最高最低温，风向 1 风向 2、风级、天气的 14 天信息文件。

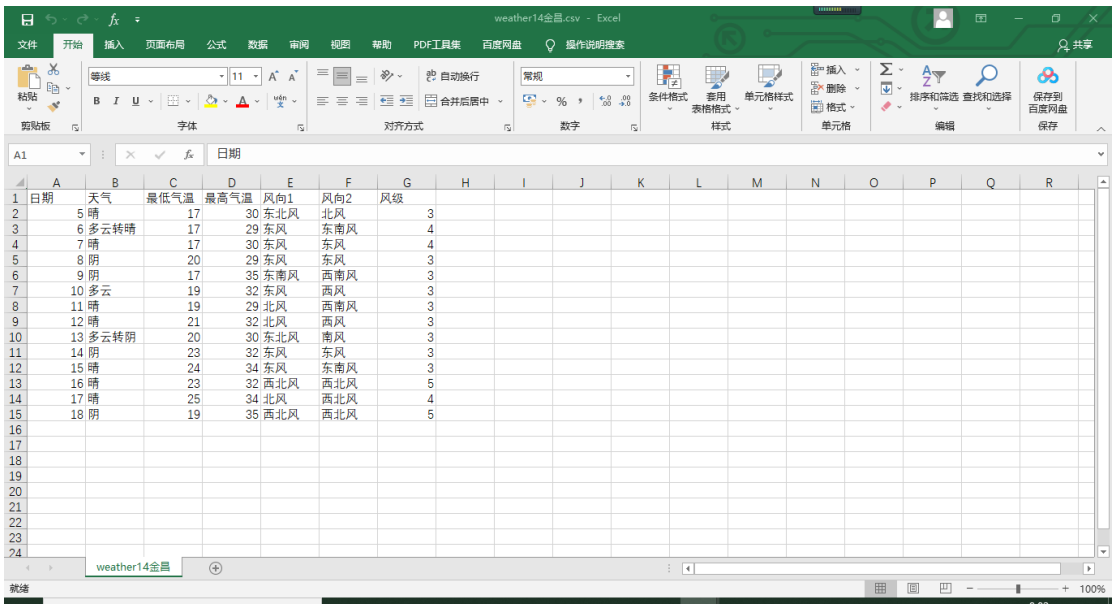
以下为成功生成的 csv 文件截图。



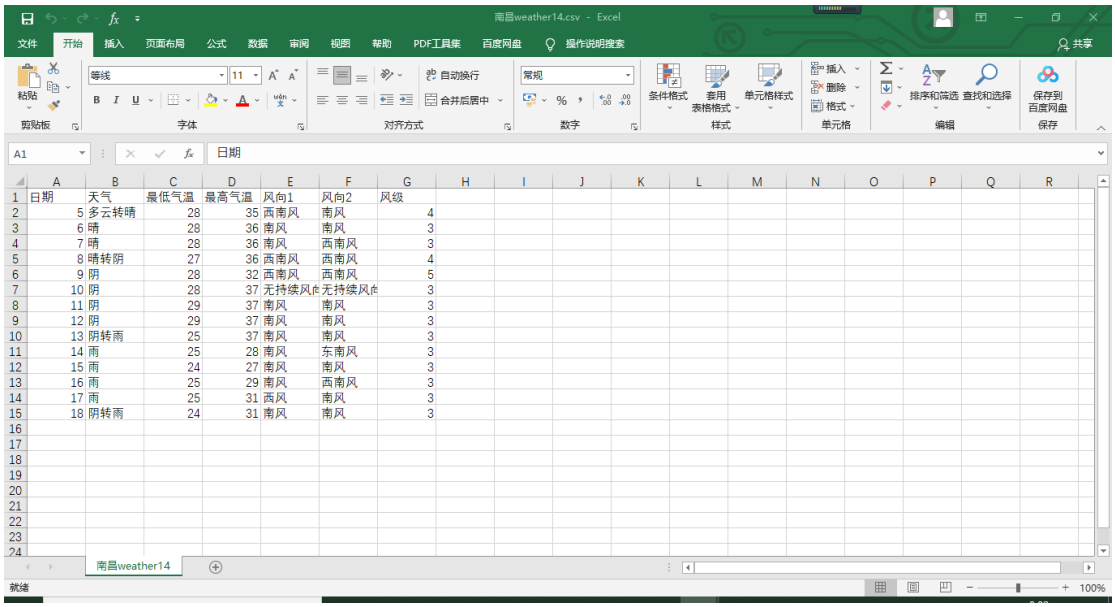
小时	温度	风力方向	风级	降水量	相对湿度	空气质量
7	21	北风	2	0	30	124
6	21	北风	2	0	30	138
5	21	西北风	1	0	28	141
4	22	西北风	1	0	30	139
3	22	西北风	1	0	33	123
2	24	西北风	2	0	26	83
1	25	西北风	2	0	24	52
0	26	西北风	2	0	22	52
23	21	西北风	1	0	36	52
22	21	北风	8	0	36	31
21	22	东南风	1	0	33	30
20	26	东风	1	0	29	38
19	32	东风	2	0	14	39
18	32	东风	2	0	15	40
17	33	东北风	2	0	14	40
16	33	东北风	3	0	12	39
15	34	东北风	2	0	13	39
14	33	东风	3	0	13	39
13	32	东北风	2	0	13	39
12	31	东南风	1	0	14	38
11	30	南风	2	0	16	53
10	28	南风	2	0	17	55
9	26	东北风	1	0	25	46
8	21	东北风	2	0	34	36



小时	温度	风力方向	风级	降水量	相对湿度	空气质量
9	28	西北风	1	0	92	
7	27	东北风	1	0	94	22
6	26	东风	1	0	97	23
5	26	东风	1	0	97	22
4	26	东风	1	0.1	97	22
3	26	东风	1	0	97	22
2	26	东北风	1	0	97	22
1	26	东风	1	0	97	22
0	26	东风	1	0	96	22
23	27	东风	1	0	96	23
22	27	东风	1	0	96	20
21	27	东风	1	0	95	19
20	27	东北风	1	0.2	94	17
19	27	西北风	1	0	95	16
18	27	西风	1	0.4	94	22
17	27	西南风	2	0.4	93	32
16	27	西风	2	6.2	93	44
15	32	西南风	2	0	72	45
14	32	西南风	2	0	72	44
13	32	西南风	3	0	72	46
12	31	西南风	2	0	72	46
11	31	西南风	2	0	72	47
10	30	西南风	3	0	75	47
9	30	西南风	2	0	77	48



日期	天气	最低气温	最高气温	风向1	风向2	风级
5	晴	17	30	东北风	北风	3
6	多云转晴	17	29	东风	东南风	4
7	晴	17	30	东风	东风	4
8	阴	20	29	东风	东风	3
9	阴	17	35	东南风	西南风	3
10	多云	19	32	东风	西风	3
11	晴	19	29	北风	西南风	3
12	晴	21	32	北风	西风	3
13	多云转阴	20	30	东北风	南风	3
14	阴	23	32	东风	东风	3
15	晴	24	34	东风	东南风	3
16	晴	23	32	西北风	西北风	5
17	晴	25	34	北风	西北风	4
18	阴	19	35	西北风	西北风	5



日期	天气	最低气温	最高气温	风向1	风向2	风级
5	多云转晴	28	35	西南风	南风	4
6	晴	28	36	南风	南风	3
7	晴	28	36	南风	西南风	3
8	晴转阴	27	36	西南风	西南风	4
9	阴	28	32	西南风	西南风	5
10	阴	28	37	无持续风向	无持续风向	3
11	阴	29	37	南风	南风	3
12	阴	29	37	南风	南风	3
13	阴转雨	25	37	南风	南风	3
14	雨	25	28	南风	东南风	3
15	雨	24	27	南风	南风	3
16	雨	25	29	南风	西南风	3
17	雨	25	31	西风	南风	3
18	阴转雨	24	31	南风	南风	3

4 数据分析图表

对 4 个 csv 文件中得到的数据做简单分析，初步判断利用这些数据可以生成风向雷达图、一天空气质量变化曲线图、温差曲线图等，但同时发现需要对数据进行进一步处理，比如利用函数设计公式求出平均温度、找出数据中的最大最小值等。

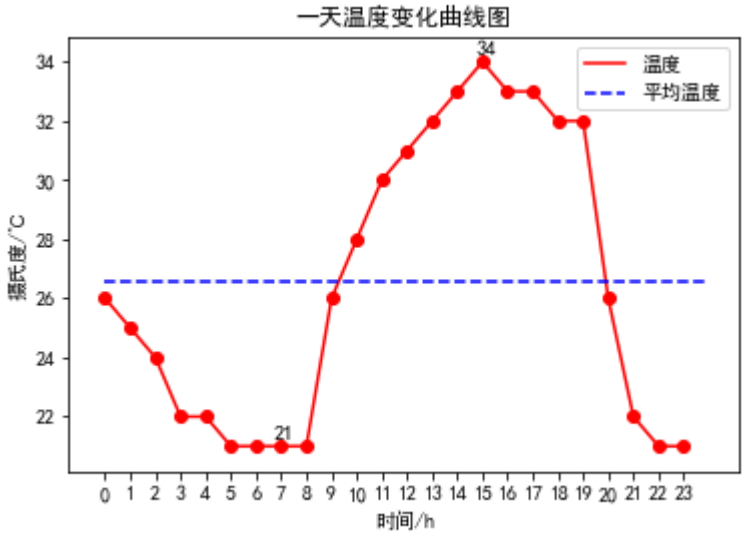
为生成可视化图表，提前求出平均温度、最高温度、最低温度、平均相对湿度、最高相对湿度、最低相对湿度、平均空气质量、最高空气质量、最低空气质量等所需要的数据，根据圆心角角度定义风向，根据风力数值设置雷达图风级。

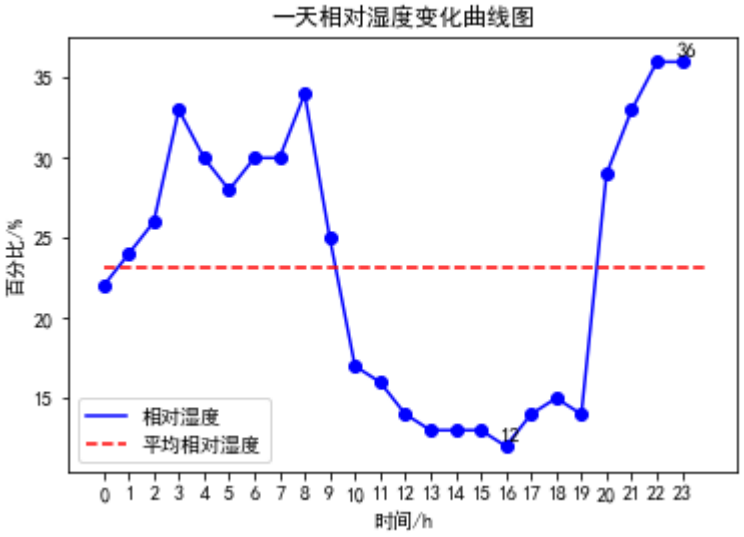
调整设置图表中的线条颜色、区域颜色等。

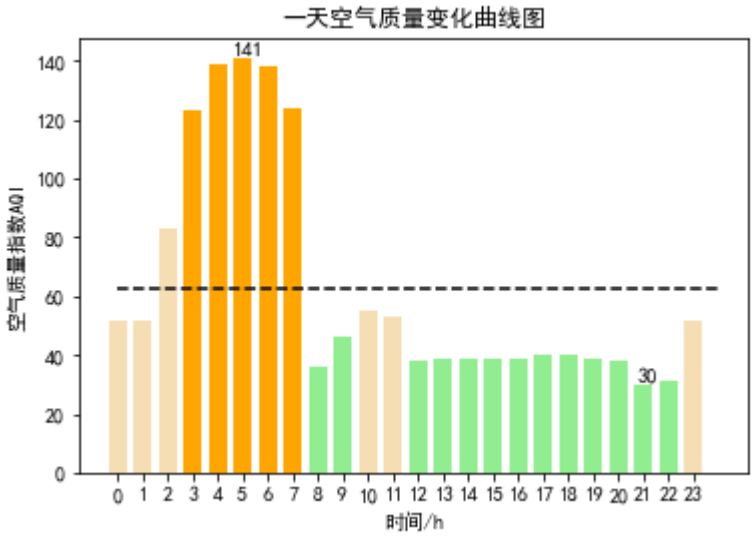
最终得出以下图表。

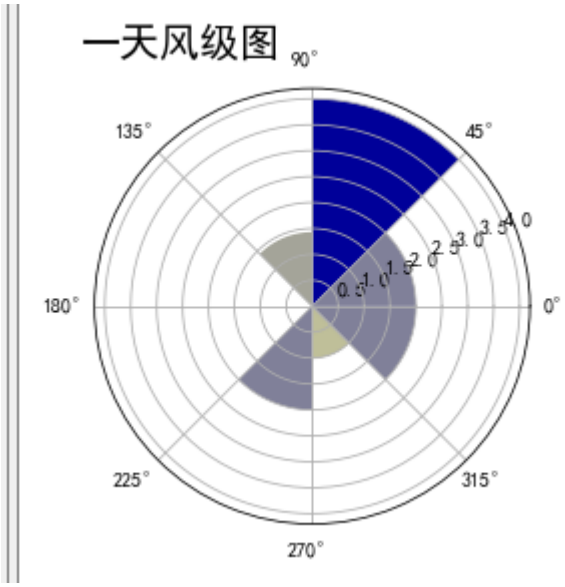
金昌市一天数据可视化

```
In [46]: runfile('C:/Users/29379/Desktop/untitled4.py', wdir='C:/Users/29379/Desktop')
Weather test
成功访问
成功访问
    小时  温度 风力方向  风级  降水量  相对湿度  空气质量
0    7  21   北风   2  0.0    30   124
1    6  21   北风   2  0.0    30   138
2    5  21  西北风   1  0.0    28   141
3    4  22  西北风   1  0.0    30   139
4    3  22  西北风   1  0.0    33   123
5    2  24  西北风   2  0.0    26    83
6    1  25  西北风   2  0.0    24    52
7    0  26  西北风   2  0.0    22    52
8   23  21  西北风   1  0.0    36    52
9   22  21   北风   8  0.0    36    31
10  21  22  东南风   1  0.0    33    30
11  20  26   东风   1  0.0    29    38
12  19  32   东风   2  0.0    14    39
13  18  32   东风   2  0.0    15    40
14  17  33  东北风   2  0.0    14    40
15  16  33  东北风   3  0.0    12    39
16  15  34  东北风   2  0.0    13    39
17  14  33   东风   3  0.0    13    39
18  13  32  东北风   2  0.0    13    39
19  12  31  东南风   1  0.0    14    38
20  11  30   南风   2  0.0    16    53
21  10  28   南风   2  0.0    17    55
22   9  26  东北风   1  0.0    25    46
23   8  21  东北风   2  0.0    34    36
```

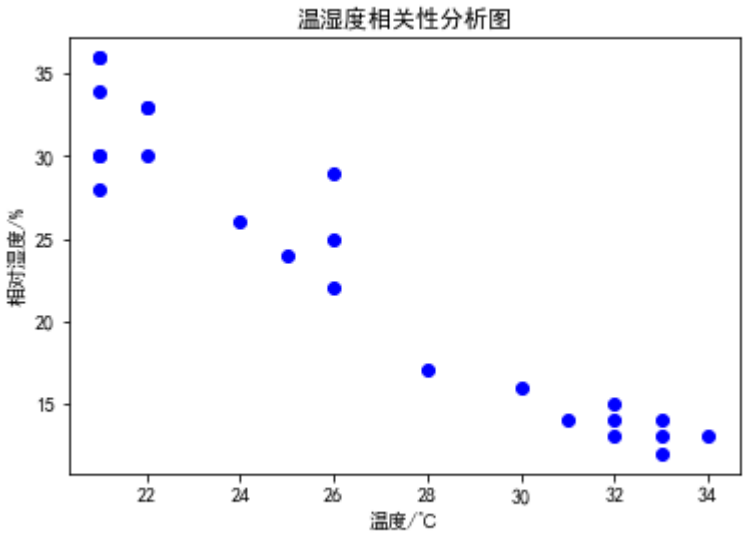

一天温度变化曲线图
温度
平均温度
摄氏度/℃
时间/h
34
21


一天相对湿度变化曲线图
相对湿度
平均相对湿度
百分比/%
时间/h
36
12


一天空气质量变化曲线图
空气质量指数AQI
时间/h
141
30



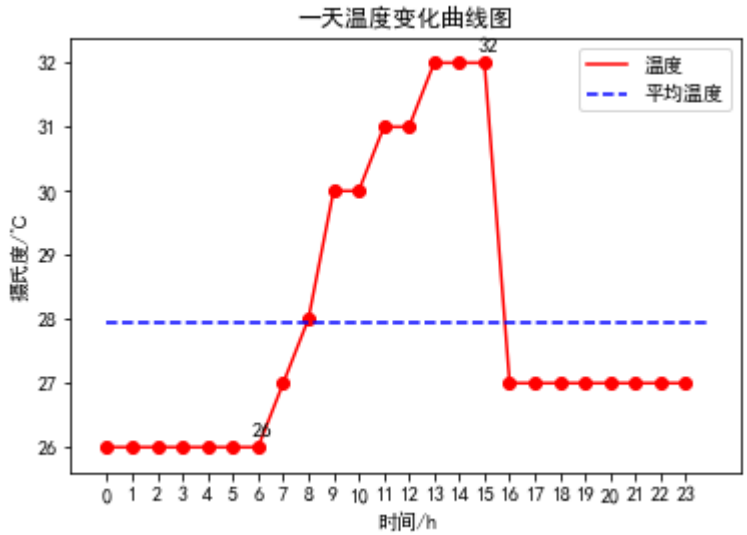
相关系数为：-0.9597251960808765

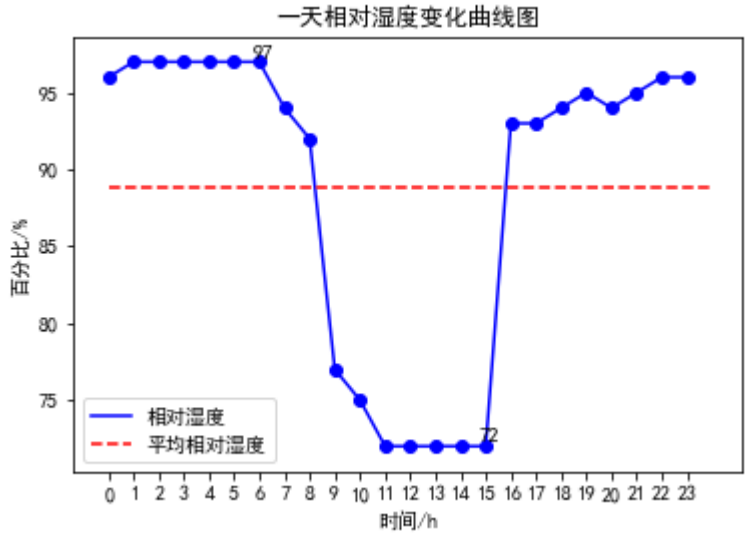


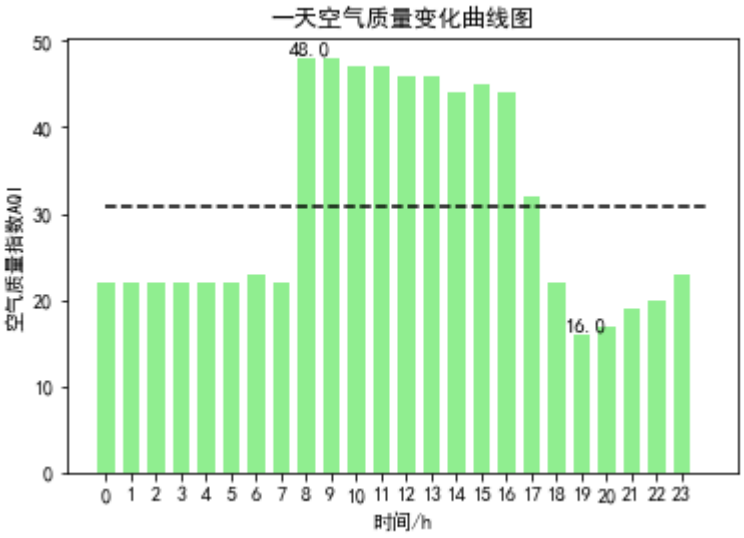
相关系数为：-0.9597251960808765

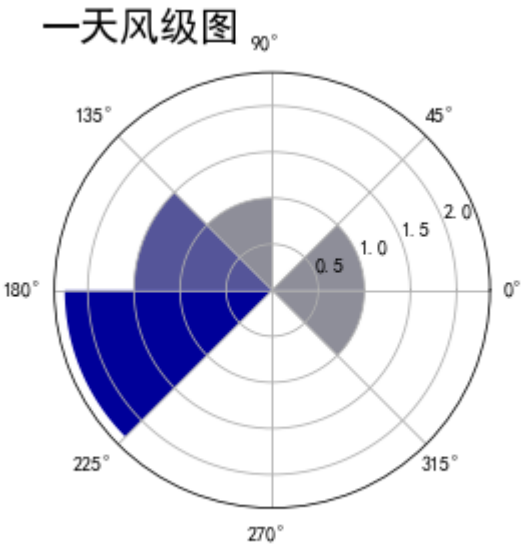
南昌市一天数据可视化

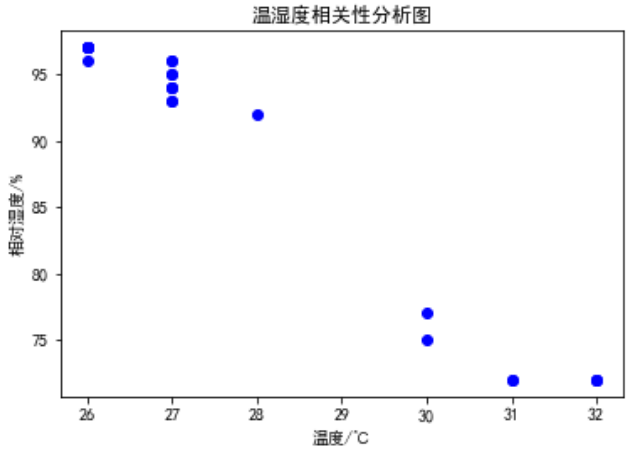
```
In [47]: runfile('C:/Users/29379/Desktop/untitled4.py', wdir='C:/Users/29379/Desktop')
Weather test
成功访问
成功访问
    小时  温度 风力方向  风级  降水量  相对湿度  空气质量
0    8  28  西北风   1  0.0   92   NaN
1    7  27  东北风   1  0.0   94  22.0
2    6  26   东风   1  0.0   97  23.0
3    5  26   东风   1  0.0   97  22.0
4    4  26   东风   1  0.1   97  22.0
5    3  26   东风   1  0.0   97  22.0
6    2  26  东北风   1  0.0   97  22.0
7    1  26   东风   1  0.0   97  22.0
8    0  26   东风   1  0.0   96  22.0
9   23  27   东风   1  0.0   96  23.0
10  22  27   东风   1  0.0   96  20.0
11  21  27   东风   1  0.0   95  19.0
12  20  27  东北风   1  0.2   94  17.0
13  19  27  西北风   1  0.0   95  16.0
14  18  27   西风   1  0.4   94  22.0
15  17  27  西南风   2  0.4   93  32.0
16  16  27   西风   2  6.2   93  44.0
17  15  32  西南风   2  0.0   72  45.0
18  14  32  西南风   2  0.0   72  44.0
19  13  32  西南风   3  0.0   72  46.0
20  12  31  西南风   2  0.0   72  46.0
21  11  31  西南风   2  0.0   72  47.0
22  10  30  西南风   3  0.0   75  47.0
23   9  30  西南风   2  0.0   77  48.0
```




一天相对湿度变化曲线图
百分比/%
时间/h
相对湿度
平均相对湿度
97
72


一天空气质量变化曲线图
空气质量指数AQI
时间/h
48.0
16.0


一天风级图
90°
45°
0°
315°
270°
225°
180°
135°
0.5
1.0
1.5
2.0



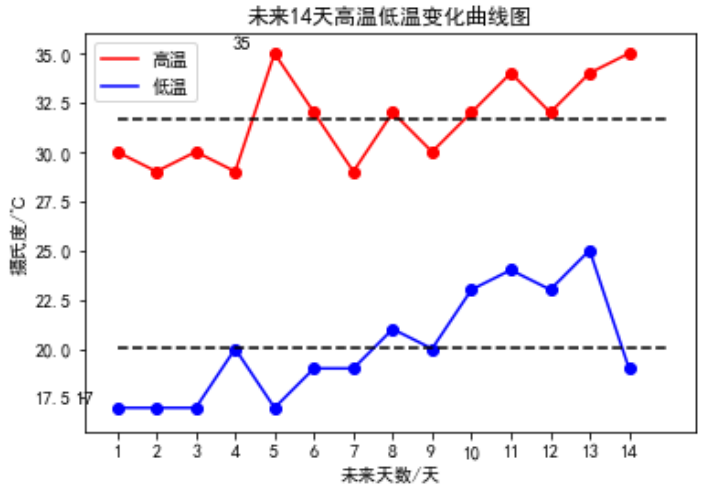
相关系数为：-0.9812761009375512

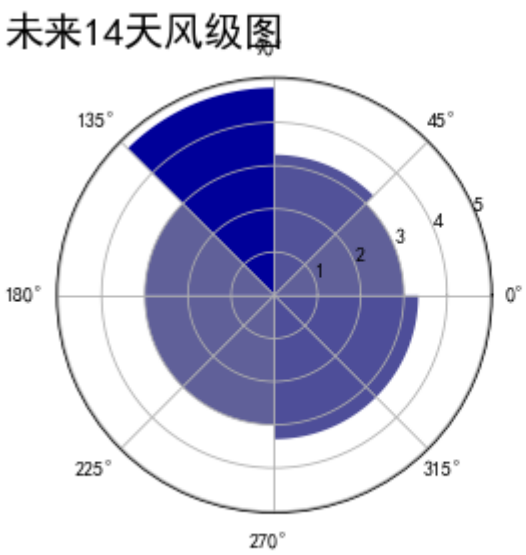
```
相关系数为：-0.9812761009375512
```

（南昌市一天数据中的温湿度相关性分析图不知道哪里出了问题有一点跑偏）

金昌市 14 天数据可视化

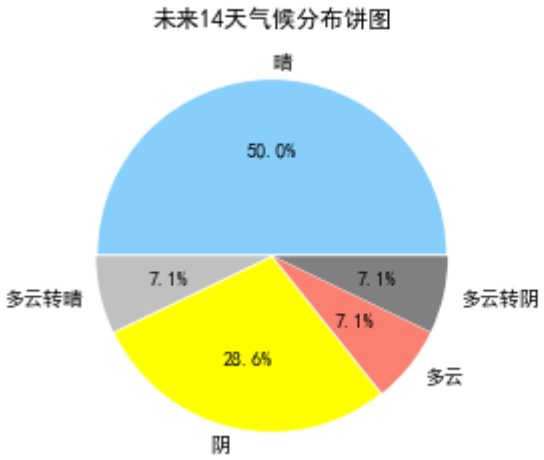
```
In [48]: runfile('C:/Users/29379/Desktop/金昌14天.py', wdir='C:/Users/29379/Desktop')
Weather test
成功访问
成功访问
```





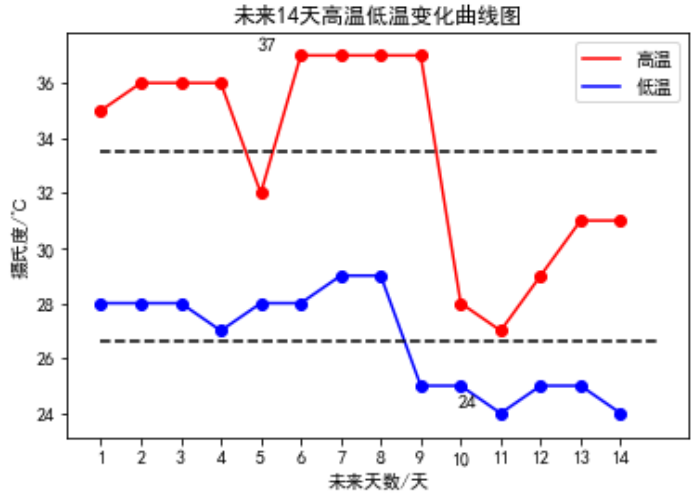
{'晴': 7, '多云转晴': 1, '阴': 4, '多云': 1, '多云转阴': 1}

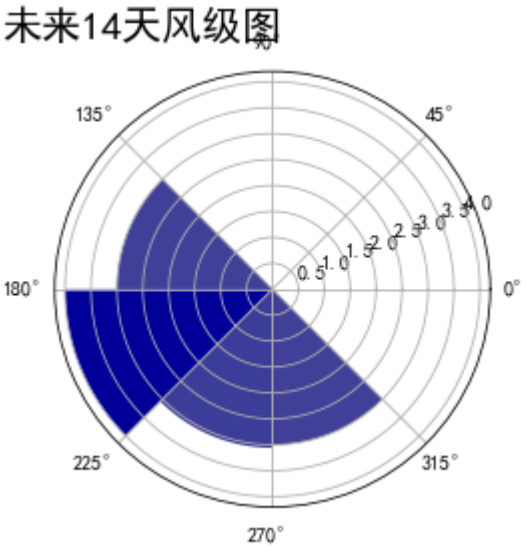
{'晴': 7, '多云转晴': 1, '阴': 4, '多云': 1, '多云转阴': 1}



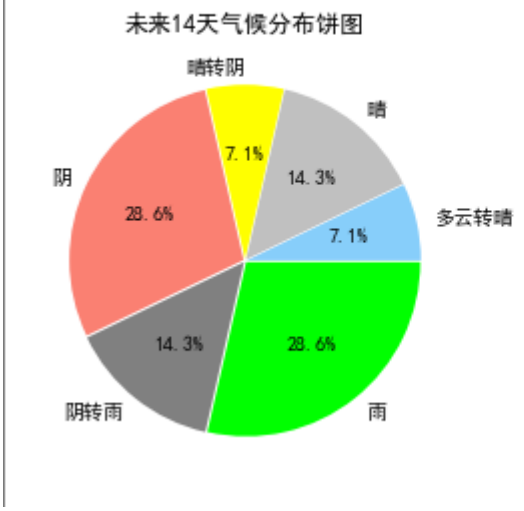
南昌市 14 天数据可视化

```
In [49]: runfile('C:/Users/29379/Desktop/南昌14.py', wdir='C:/Users/29379/Desktop')
Weather test
成功访问
成功访问
```





```
{'多云转晴': 1, '晴': 2, '晴转阴': 1, '阴': 4, '阴转雨': 2, '雨': 4}
```



5 数据分析结果

基本结论综述

由可视化结果可知，金昌市当天最高温度 34℃，最低温度 21℃，当日均温 27℃，14、15 时达到最高值，气温从 19 时开始骤降，于 8 时前的 4、5、6、7 时达到最低值。一天相对湿度平均值不到 25%，10-19 时时湿度最低，凌晨到 7 时左右空气污染非常严重。风力不强，最大 3 级，多为西北风和东北风。

南昌市当天最高温度 32℃，最低温度 26℃，平均相对湿度接近 90%，相对湿度最高值有 97%且高湿度持续时间长。全天空气质量都很好，但 8-16 时空气质量相对差。风力弱，多为偏东风和西南风，全天风力低于 2 级。

从当天数据分析结果，可以看到如资料中所述：金昌市空气干燥，昼夜温差大，空气污染相对严重。而南昌市当天温差较小，空气质量更高，但相对湿度非常高，空气潮湿。那么，将当日数据分析结果扩展至 14 日时，基本结论会保持一致吗？

由未来 14 天数据可知，金昌市平均高温 32℃左右，平均低温 20℃左右；南昌市平均高温 33℃，平均低温 27℃，但南昌市高低温变化曲线图明显展现出强烈波动，有 4 天可达到 37 度高温，但将于 7 月 10 日骤降至 28 度。金昌市未来 14 天内风力最高达到 5 级，南昌市最高达到 4 级。未来 14 天内，金昌市晴天 7 天，其他天气 7 天，不下雨；而南昌市阴、阴转雨、雨天概率就达到 71.5%，只有两天晴天。

对比 14 天两地数据可知，其反映出的结果基本与当日所反映出的结果没有较大差别。金昌市气温的特点为昼夜温差大、夜晚温度低，也由于大部分晴天和较高的海拔，金昌市每日不同时段的温度变化十分明显，所记录 7 月间基本每日 20：40 左右日落，因此此时气温骤降，不过这一特征在夏天的夜晚给人的感觉一般是凉爽，但也需要注意早晚出门准备一件薄外套。同时，在干燥的环境中居住，人们要注意饮食、避免上火，更要注意夏季防火、随时关注身边的易燃物。金昌市位于中国西北部地区，干旱的环境和稀少的植被更导致易起沙尘，而其作为典型的工业城市，工业对空气的污染是不可避免的，因此导致空气质量较差，夜晚出现的逆温层又会更加剧夜晚的空气污染程度，造成数据中表现出的现象。

南昌市位于中国东南部地区，季风气候导致 7 月正是夏季多雨期，14 天内只出现 2 天晴天的现象也不奇怪，但伴随而来的是空气的高湿度，衣物不干甚至发霉、物品受潮等现象

容易发生。值得关注的是连续 4 天达到 37℃高温后的骤降至 28℃，其实不难推测出是由于高温几天多为晴天导致的温度攀升与阴天不断聚集的大气温度，而后开始降雨，这种天气情况容易引发强对流，未来半个月生活在南昌的人们应及时关注天气预报、随身携带雨伞、避免在雷雨天气出门。

两市数据中还有一个数据图表名为温湿度相关性分析，根据每日不同时段的温度和相对湿度计算得知，其结果分别为金昌市-0.95，南昌市-0.98，也就是意味着两地的温湿度都表现出极强的负相关关系，这就是说，每日温度高时相对湿度便会有一定下降，南昌市的人们可以在温度最高的时段选择晾晒衣物、谨防发霉。

数据结果思考与启示

我个人由于西北地区的干旱和稀少的降水量，自认为比较喜欢阴雨天气，但在来南昌读书的接近一年间逐渐对阴雨天产生了烦恼情绪，衣物、床品的潮湿与发霉、天花板与地面的回潮这些无论放多少防潮物品都无法解决的问题使我非常苦恼，同时潮热的空气容易致使皮肤粘腻、出油，不论白天夜晚的高温使我失去出门的兴趣，想必 7 月及暑假期间能够选择居住地，我不会选择留住在南昌。

金昌市目前被评定为五线城市，经济发展水平与繁华程度定然不比南昌市这样一座省会城市。它展现出典型的冬冷夏热、昼夜温差大、多大风沙尘等大陆性特征，7 月及暑假期间由于高温和空气质量，金昌市民们多多少少都会犯季节性鼻炎，这也是无法解决的问题之一。然而金昌市全年降水量少，夏季依旧以晴天为主，夏天的夜晚有凉爽干燥的空气、深邃幽蓝的天空、一望无垠的戈壁和满眼的星辰，令我深深沉迷、不能自拔。

湿度、温度、风……每一个地理因素都受众多其他因素影响，比如它们间互相影响，又比如维度、气压、逆温现象、海拔、海陆位置……天气是某一地区距离地表较近的大气层在短时间内的具体状态，不同于气候却也能反映出一地的气候特征，温带大陆性气候与亚热带季风气候定然有千差万别，这正是中国大地上的神奇的自然与地理，值得我们毕生观察与研究。

6 总结

参考数据与文章

本数据分析报告作业爬取中国天气网数据，并在CSDN 中研究与参考了这七篇博文，感谢这些博主写出的关于天气数据分析报告、数据可视化、爬虫相关的博文，帮助我成功完成了本次报告。同时本人保证本次作业为本人独立完成。

数据分析报告制作过程、收获与体会

首先根据所给选题确定了一个本人比较感兴趣的选题——天气数据分析，其次确立分析方向为对比家乡金昌市与现居住地南昌市，进入中国天气网观察所需要数据。在观察数据与资料查找中我发现很难直接得到某地天气数据的表格文件，只能通过爬取中国天气网获得所需要数据，此时进入 CSDN 查找了一些爬虫的资料，开始尝试对中国天气网进行爬虫。

爬虫过程非常艰难，由于不熟悉操作与页面，一开始很难找到所需要的数据对应的特征标签，找到的特征标签提取出来生成的文件也总是一片空白，我开始多方面查找针对中国天气网的爬虫资料，终于摸索出了爬取当日和 8-15 天天气数据。其实我一开始定题为“暑假期间我更适合在哪居住”，试想的分析范围至少在未来一个月及以上，但由于中国天气网 40 天天气数据排版更为复杂，我没有研究透彻如何爬取只好作罢，也算是本次报告的一个遗憾。

要进行可视化，就要写出对图像的很多设置条件，在此过程中我遇到了相当多困难，博主给出的似乎没有问题的代码接连报错、运行时偶然成功一次后再也没有成功、想不到计算公式、程序运行至空数据时又报错……接连的以前没有见过的错误使我对语句语法和程序运行有了更深入的了解。下面进行简单举例：

TypeError：must be real number, not str 输入的值必须是实数，不是字符串，也就是说这里读到了字符串，这句的目的是找到空值 nan。查阅资料后得出多种解决方法，逐次尝试最终改变了这条语句的写法然后纠正了这个错误。

运行时偶然成功一次之后再也无法成功。这个问题报的错由于搜索记录找不到而丢失了，这里通过搜索也找不到问题所在，戏剧性地因为要清理桌面上生成的全部数据文件后又运行了一次居然成功了而解决，原来是成功生成后的文件影响了第二次运行的数据提取与计算，因此清除了前面的运行结果就可以成功运行了。

总之，本次报告来之不易，连续长时间的实战和研究使我发现在学习 python 这门课程时，不可以只流于表面，书本上的代码永远解决不了实际问题，必须通过亲身体会报错和查找原因才能在认知上得到较大的提升。数据可视化的图表在生成好的那一刻，成功的满足感和惊喜感带给我不一样的感受，此前我从没有单独解决过大批量数据的可视化问题，这次过程同样让我看到了 python 功能的强大和数据可视化的重要性。我想如果以后可以继续坚持学习 python，不失为一种明智的选择，希望我可以保持这份好奇心与热情。

附录-数据分析代码

提前说明：
①利用中国天气网查找某一城市的天气信息时，只有城市代码号会发生改变，但研究 1 天与 8-15 天的天气信息时网址会发生改变。因此以下代码运行时，为防止出错，只需要改变网址代码里的城市代码号。
②代码需要重复运行时，必须记录好此前的数据并对代码运行后保存的几个文件进行删除，否则之前运行出的数据会对此后的数据造成影响，容易报错和造成数值有误。

1.金昌 1 天

```
# -*- coding: utf-8 -*-
"""
Created on Sun Jul    4 02:26:08 2021

@author: 29379
"""

# weather.py
import requests
from bs4 import BeautifulSoup
import csv
import json
def getHTMLtext(url):
    """请求获得网页内容"""
    try:
        r = requests.get(url, timeout = 30)
        r.raise_for_status()
        r.encoding = r.apparent_encoding
        print("成功访问")
        return r.text
    except:
```

```
        print("访问错误")
        return" "

def get_content(html):
    """处理得到有用信息保存数据文件"""
    final = []                                  # 初始化一个列表保存数据
    bs = BeautifulSoup(html, "html.parser")   # 创建 BeautifulSoup 对象
    body = bs.body
    data = body.find('div', {'id': '7d'})       # 找到 div 标签且 id = 7d
    # 下面爬取当天的数据
    data2 = body.find_all('div',{'class':'left-div'})
    text = data2[2].find('script').string
    text = text[text.index('=')+1 :-2]          # 移除改 var data=将其变为 json 数据
    jd = json.loads(text)
    dayone = jd['od']['od2']                    # 找到当天的数据
    final_day = []                                  # 存放当天的数据
    count = 0
    for i in dayone:
        temp = []
        if count <=23:
            temp.append(i['od21'])                  # 添加时间
            temp.append(i['od22'])                  # 添加当前时刻温度
            temp.append(i['od24'])                  # 添加当前时刻风力方向
            temp.append(i['od25'])                  # 添加当前时刻风级
            temp.append(i['od26'])                  # 添加当前时刻降水量
            temp.append(i['od27'])                  # 添加当前时刻相对湿度
            temp.append(i['od28'])                  # 添加当前时刻控制质量
            #print(temp)
            final_day.append(temp)
        count = count +1
    # 下面爬取 7 天的数据
    ul = data.find('ul')                        # 找到所有的 ul 标签
    li = ul.find_all('li')                      # 找到左右的 li 标签
    i = 0                   # 控制爬取的天数
    for day in li:                                  # 遍历找到的每一个 li
        if i < 7 and i > 0:
            temp = []                                       # 临时存放每天的数据
            date = day.find('h1').string            # 得到日期
            date = date[0:date.index('日')]      # 取出日期号
            temp.append(date)
            inf = day.find_all('p')                 # 找出 li 下面的 p 标签,提取第一个 p 标签的
值，即天气
            temp.append(inf[0].string)
```

```
                tem_low = inf[1].find('i').string     # 找到最低气温

                if inf[1].find('span') is None:       # 天气预报可能没有最高气温
                    tem_high = None
                else:
                    tem_high = inf[1].find('span').string   # 找到最高气温
                temp.append(tem_low[:-1])
                if tem_high[-1] == '℃':
                    temp.append(tem_high[:-1])
                else:
                    temp.append(tem_high)

                wind = inf[2].find_all('span')        # 找到风向
                for j in wind:
                    temp.append(j['title'])

                wind_scale = inf[2].find('i').string # 找到风级
                index1 = wind_scale.index('级')
                temp.append(int(wind_scale[index1-1:index1]))
                final.append(temp)
            i = i + 1
    return final_day,final
    #print(final)
def get_content2(html):
    """处理得到有用信息保存数据文件"""
    final = []                                                  # 初始化一个列
表保存数据
    bs = BeautifulSoup(html, "html.parser")                     # 创建 BeautifulSoup
对象
    body = bs.body
    data = body.find('div', {'id': '15d'})                     # 找到 div 标签且 id = 15d
    ul = data.find('ul')                                        # 找到所有的 ul 标签
    li = ul.find_all('li')                                      # 找到左右的 li 标签
    final = []
    i = 0                                                       # 控制爬取
的天数
    for day in li:                                              # 遍历找到的每一个
li
        if i < 8:
            temp = []                                           # 临时存放
每天的数据
            date = day.find('span',{'class':'time'}).string     # 得到日期
            date = date[date.index('（')+1:-2]                  # 取出日期号
            temp.append(date)
```

```
            weather = day.find('span',{'class':'wea'}).string          # 找到天气
            temp.append(weather)
            tem = day.find('span',{'class':'tem'}).text                # 找到温度
            temp.append(tem[tem.index('/')+1:-1])                     # 找到最低气温
            temp.append(tem[:tem.index('/')-1])                       # 找到最高气温
            wind = day.find('span',{'class':'wind'}).string           # 找到风向
            if '转' in wind:                                          # 如果有风向变化
                temp.append(wind[:wind.index('转')])
                temp.append(wind[wind.index('转')+1:])
            else:                                                     # 如果没有风向变化，前后风向一致
                temp.append(wind)
                temp.append(wind)
            wind_scale = day.find('span',{'class':'wind1'}).string    # 找到风级
            index1 = wind_scale.index('级')
            temp.append(int(wind_scale[index1-1:index1]))

            final.append(temp)
    return final

def write_to_csv(file_name, data, day=14):
    """保存为 csv 文件"""
    with open(file_name, 'a', errors='ignore', newline='') as f:
        if day == 14:
            header = ['日期','天气','最低气温','最高气温','风向 1','风向 2','风级']
        else:
            header = ['小时','温度','风力方向','风级','降水量','相对湿度','空气质量']
        f_csv = csv.writer(f)
        f_csv.writerow(header)
        f_csv.writerows(data)

def main():
    """主函数"""
    print("Weather test")
    # 金昌
    url1 = 'http://www.weather.com.cn/weather/101160601.shtml'      # 7 天天气中国天气网
    url2 = 'http://www.weather.com.cn/weather15d/101160601.shtml' # 8-15 天天气中国天气网

    html1 = getHTMLtext(url1)
    data1, data1_7 = get_content(html1)          # 获得 1-7 天和当天的数据

    html2 = getHTMLtext(url2)
    data8_14 = get_content2(html2)               # 获得 8-14 天数据
```

```
    data14 = data1_7 + data8_14
    #print(data)
    write_to_csv('weather14.csv',data14,14)      # 保存为 csv 文件
    write_to_csv('weather1.csv',data1,1)

if __name__ == '__main__':
    main()

# data1_analysis.py
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import math
def tem_curve(data):
    """温度曲线绘制"""
    hour = list(data['小时'])
    tem = list(data['温度'])
    for i in range(0,24):
        if tem[i] == 'nan':
            tem[i] = tem[i-1]
    tem_ave = sum(tem)/24                          # 求平均温度
    tem_max = max(tem)
    tem_max_hour = hour[tem.index(tem_max)]# 求最高温度
    tem_min = min(tem)
    tem_min_hour = hour[tem.index(tem_min)] # 求最低温度
    x = []
    y = []
    for i in range(0, 24):
        x.append(i)
        y.append(tem[hour.index(i)])
    plt.figure(1)
    plt.plot(x,y,color='red',label='温度')                                    # 画出温度曲线
    plt.scatter(x,y,color='red')                # 点出每个时刻的温度点
    plt.plot([0, 24], [tem_ave, tem_ave], c='blue', linestyle='--',label='平均温度')  # 画出平均
温度虚线
    plt.text(tem_max_hour+0.15, tem_max+0.15, str(tem_max), ha='center', va='bottom',
fontsize=10.5)  # 标出最高温度
    plt.text(tem_min_hour+0.15, tem_min+0.15, str(tem_min), ha='center', va='bottom',
fontsize=10.5)  # 标出最低温度
    plt.xticks(x)
    plt.legend()
    plt.title('一天温度变化曲线图')
    plt.xlabel('时间/h')
    plt.ylabel('摄氏度/℃')
```

```
    plt.show()

def hum_curve(data):
    """相对湿度曲线绘制"""
    hour = list(data['小时'])
    hum = list(data['相对湿度'])
    for i in range(0,24):
        if math.isnan(hum[i]) == True:
            hum[i] = hum[i-1]
    hum_ave = sum(hum)/24                          # 求平均相对湿度
    hum_max = max(hum)
    hum_max_hour = hour[hum.index(hum_max)]    # 求最高相对湿度
    hum_min = min(hum)
    hum_min_hour = hour[hum.index(hum_min)]    # 求最低相对湿度
    x = []
    y = []
    for i in range(0, 24):
        x.append(i)
        y.append(hum[hour.index(i)])
    plt.figure(2)
    plt.plot(x,y,color='blue',label='相对湿度')                                    # 画出相对湿度
曲线
    plt.scatter(x,y,color='blue')                # 点出每个时刻的相对湿度
    plt.plot([0, 24], [hum_ave, hum_ave], c='red', linestyle='--',label='平均相对湿度')  # 画出
平均相对湿度虚线
    plt.text(hum_max_hour+0.15, hum_max+0.15, str(hum_max), ha='center', va='bottom',
fontsize=10.5)  # 标出最高相对湿度
    plt.text(hum_min_hour+0.15, hum_min+0.15, str(hum_min), ha='center', va='bottom',
fontsize=10.5)  # 标出最低相对湿度
    plt.xticks(x)
    plt.legend()
    plt.title('一天相对湿度变化曲线图')
    plt.xlabel('时间/h')
    plt.ylabel('百分比/%')
    plt.show()

def air_curve(data):
    """空气质量曲线绘制"""
    hour = list(data['小时'])
    air = list(data['空气质量'])
    print(type(air[0]))
    for i in range(0,24):
        if math.isnan(air[i]) == True:
            air[i] = air[i-1]
```

```
    air_ave = sum(air)/24                          # 求平均空气质量
    air_max = max(air)
    air_max_hour = hour[air.index(air_max)]    # 求最高空气质量
    air_min = min(air)
    air_min_hour = hour[air.index(air_min)]    # 求最低空气质量
    x = []
    y = []
    for i in range(0, 24):
        x.append(i)
        y.append(air[hour.index(i)])
    plt.figure(3)

    for i in range(0,24):
        if y[i] <= 50:
            plt.bar(x[i],y[i],color='lightgreen',width=0.7)   # 1 等级
        elif y[i] <= 100:
            plt.bar(x[i],y[i],color='wheat',width=0.7)            # 2 等级
        elif y[i] <= 150:
            plt.bar(x[i],y[i],color='orange',width=0.7)           # 3 等级
        elif y[i] <= 200:
            plt.bar(x[i],y[i],color='orangered',width=0.7)  # 4 等级
        elif y[i] <= 300:
            plt.bar(x[i],y[i],color='darkviolet',width=0.7)       # 5 等级
        elif y[i] > 300:
            plt.bar(x[i],y[i],color='maroon',width=0.7)           # 6 等级
    plt.plot([0, 24], [air_ave, air_ave], c='black', linestyle='--')  # 画出平均空气质量虚线
    plt.text(air_max_hour+0.15,    air_max+0.15,    str(air_max),    ha='center',    va='bottom',
fontsize=10.5)  # 标出最高空气质量
    plt.text(air_min_hour+0.15,    air_min+0.15,    str(air_min),    ha='center',    va='bottom',
fontsize=10.5)  # 标出最低空气质量
    plt.xticks(x)
    plt.title('一天空气质量变化曲线图')
    plt.xlabel('时间/h')
    plt.ylabel('空气质量指数 AQI')
    plt.show()

def wind_radar(data):
    """风向雷达图"""
    wind = list(data['风力方向'])
    wind_speed = list(data['风级'])
    for i in range(0,24):
        if wind[i] == "北风":
            wind[i] = 90
        elif wind[i] == "南风":
```

```
                wind[i] = 270
            elif wind[i] == "西风":
                wind[i] = 180
            elif wind[i] == "东风":
                wind[i] = 360
            elif wind[i] == "东北风":
                wind[i] = 45
            elif wind[i] == "西北风":
                wind[i] = 135
            elif wind[i] == "西南风":
                wind[i] = 225
            elif wind[i] == "东南风":
                wind[i] = 315
        degs = np.arange(45,361,45)
        temp = []
        for deg in degs:
            speed = []
            # 获取 wind_deg 在指定范围的风速平均值数据
            for i in range(0,24):
                if wind[i] == deg:
                    speed.append(wind_speed[i])
            if len(speed) == 0:
                temp.append(0)
            else:
                temp.append(sum(speed)/len(speed))
        print(temp)
        N = 8
        theta = np.arange(0.+np.pi/8,2*np.pi+np.pi/8,2*np.pi/8)
        # 数据极径
        radii = np.array(temp)
        # 绘制极区图坐标系
        plt.axes(polar=True)
        # 定义每个扇区的 RGB 值（R,G,B），x 越大，对应的颜色越接近蓝色
        colors = [(1-x/max(temp), 1-x/max(temp),0.6) for x in radii]
        plt.bar(theta,radii,width=(2*np.pi/N),bottom=0.0,color=colors)
        plt.title('一天风级图',x=0.2,fontsize=20)
        plt.show()

def calc_corr(a, b):
        """计算相关系数"""
        a_avg = sum(a)/len(a)
        b_avg = sum(b)/len(b)
        cov_ab = sum([(x - a_avg)*(y - b_avg) for x,y in zip(a, b)])
        sq = math.sqrt(sum([(x - a_avg)**2 for x in a])*sum([(x - b_avg)**2 for x in b]))
```

```
    corr_factor = cov_ab/sq
    return corr_factor


def corr_tem_hum(data):
    """温湿度相关性分析"""
    tem = data['温度']
    hum = data['相对湿度']
    plt.scatter(tem,hum,color='blue')
    plt.title("温湿度相关性分析图")
    plt.xlabel("温度/℃")
    plt.ylabel("相对湿度/%")
    plt.text(20,40,"相关系数为："+str(calc_corr(tem,hum)),fontdict={'size':'10','color':'red'})
    plt.show()
    print("相关系数为："+str(calc_corr(tem,hum)))

def main():
    plt.rcParams['font.sans-serif']=['SimHei']     # 解决中文显示问题
    plt.rcParams['axes.unicode_minus'] = False   # 解决负号显示问题
    data1 = pd.read_csv('weather1.csv',encoding='gb2312')
    print(data1)
    tem_curve(data1)
    hum_curve(data1)
    air_curve(data1)
    wind_radar(data1)
    corr_tem_hum(data1)
if __name__ == '__main__':
    main()
```

2.南昌 1 天

```
# -*- coding: utf-8 -*-
"""
Created on Sun Jul  4 08:38:57 2021

@author: 29379
"""

# data14_analysis.py

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import math
```

```
def tem_curve(data):
    """温度曲线绘制"""
    date = list(data['日期'])
    tem_low = list(data['最低气温'])
    tem_high = list(data['最高气温'])
    for i in range(0, 14):
        if math.isnan(tem_low[i]) == True:
            tem_low[i] = tem_low[i - 1]
        if math.isnan(tem_high[i]) == True:
            tem_high[i] = tem_high[i - 1]

    tem_high_ave = sum(tem_high) / 14  # 求平均高温
    tem_low_ave = sum(tem_low) / 14  # 求平均低温

    tem_max = max(tem_high)
    tem_max_date = tem_high.index(tem_max)  # 求最高温度
    tem_min = min(tem_low)
    tem_min_date = tem_low.index(tem_min)  # 求最低温度

    x = range(1, 15)
    plt.figure(1)
    plt.plot(x, tem_high, color='red', label='高温')  # 画出高温度曲线
    plt.scatter(x, tem_high, color='red')  # 点出每个时刻的温度点
    plt.plot(x, tem_low, color='blue', label='低温')  # 画出低温度曲线
    plt.scatter(x, tem_low, color='blue')  # 点出每个时刻的温度点

    plt.plot([1, 15], [tem_high_ave, tem_high_ave], c='black', linestyle='--')  # 画出平均温度虚
线
    plt.plot([1, 15], [tem_low_ave, tem_low_ave], c='black', linestyle='--')  # 画出平均温度虚
线
    plt.legend()
    plt.text(tem_max_date + 0.15, tem_max + 0.15, str(tem_max), ha='center', va='bottom',
fontsize=10.5)  # 标出最高温度
    plt.text(tem_min_date + 0.15, tem_min + 0.15, str(tem_min), ha='center', va='bottom',
fontsize=10.5)  # 标出最低温度
    plt.xticks(x)
    plt.title('未来 14 天高温低温变化曲线图')
    plt.xlabel('未来天数/天')
    plt.ylabel('摄氏度/℃')
    plt.show()

def change_wind(wind):
    """改变风向"""
```

```
    x = ["北风","南风","西风","东风","东北风","西北风","西南风","东南风"]
    y = [90,270,180,360,45,135,225,315]
    for i in range(0,14):
        for j in range(len(x)):
            if wind[i] == x[j]:
                wind[i] = y[j]
    return wind

def wind_radar(data):
    """风向雷达图"""
    wind1 = list(data['风向 1'])
    wind2 = list(data['风向 2'])
    wind_speed = list(data['风级'])
    wind1 = change_wind(wind1)
    wind2 = change_wind(wind2)

    degs = np.arange(45, 361, 45)
    temp = []
    for deg in degs:
        speed = []
        # 获取 wind_deg 在指定范围的风速平均值数据
        for i in range(0, 14):
            if wind1[i] == deg:
                speed.append(wind_speed[i])
            if wind2[i] == deg:
                speed.append(wind_speed[i])
        if len(speed) == 0:
            temp.append(0)
        else:
            temp.append(sum(speed) / len(speed))
    print(temp)
    N = 8
    theta = np.arange(0. + np.pi / 8, 2 * np.pi + np.pi / 8, 2 * np.pi / 8)
    # 数据极径
    radii = np.array(temp)
    # 绘制极区图坐标系
    plt.axes(polar=True)
    # 定义每个扇区的 RGB 值（R,G,B），x 越大，对应的颜色越接近蓝色
    colors = [(1 - x / max(temp), 1 - x / max(temp), 0.6) for x in radii]
    plt.bar(theta, radii, width=(2 * np.pi / N), bottom=0.0, color=colors)
    plt.title('未来 14 天风级图', x=0.2, fontsize=20)
    plt.show()

def weather_pie(data):
```

```
    """绘制天气饼图"""
    weather = list(data['天气'])
    dic_wea = {}
    for i in range(0, 14):
        if weather[i] in dic_wea.keys():
            dic_wea[weather[i]] += 1
        else:
            dic_wea[weather[i]] = 1
    print(dic_wea)
    explode = [0.01] * len(dic_wea.keys())
    color = ['lightskyblue', 'silver', 'yellow', 'salmon', 'grey', 'lime', 'gold', 'red', 'green', 'pink']
    plt.pie(dic_wea.values(), explode=explode, labels=dic_wea.keys(), autopct='%1.1f%%',
colors=color)
    plt.title('未来 14 天气候分布饼图')
    plt.show()

def main():
    plt.rcParams['font.sans-serif'] = ['SimHei']  # 解决中文显示问题
    plt.rcParams['axes.unicode_minus'] = False  # 解决负号显示问题
    data14 = pd.read_csv('weather14.csv', encoding='gb2312')
    # print(data14)
    tem_curve(data14)#未来 14 天高低温度变化
    wind_radar(data14)#未来 14 天风级图
    weather_pie(data14)#未来 14 天气候分布饼图

if __name__ == '__main__':
    main()
```

3.金昌 14 天

```
# -*- coding: utf-8 -*-
"""
Created on Sun Jul  4 08:51:31 2021

@author: 29379
"""
# weather.py
import requests
from bs4 import BeautifulSoup
import csv
import json
def getHTMLtext(url):
    """请求获得网页内容"""
    try:
```

```
            r = requests.get(url, timeout = 30)
            r.raise_for_status()
            r.encoding = r.apparent_encoding
            print("成功访问")
            return r.text
        except:
            print("访问错误")
            return" "

def get_content(html):
    """处理得到有用信息保存数据文件"""
    final = []                                  # 初始化一个列表保存数据
    bs = BeautifulSoup(html, "html.parser")   # 创建 BeautifulSoup 对象
    body = bs.body
    data = body.find('div', {'id': '7d'})       # 找到 div 标签且 id = 7d
    # 下面爬取当天的数据
    data2 = body.find_all('div',{'class':'left-div'})
    text = data2[2].find('script').string
    text = text[text.index('=')+1 :-2]          # 移除改 var data=将其变为 json 数据
    jd = json.loads(text)
    dayone = jd['od']['od2']                    # 找到当天的数据
    final_day = []                                  # 存放当天的数据
    count = 0
    for i in dayone:
        temp = []
        if count <=23:
            temp.append(i['od21'])                  # 添加时间
            temp.append(i['od22'])                  # 添加当前时刻温度
            temp.append(i['od24'])                  # 添加当前时刻风力方向
            temp.append(i['od25'])                  # 添加当前时刻风级
            temp.append(i['od26'])                  # 添加当前时刻降水量
            temp.append(i['od27'])                  # 添加当前时刻相对湿度
            temp.append(i['od28'])                  # 添加当前时刻控制质量
            #print(temp)
            final_day.append(temp)
        count = count +1
    # 下面爬取 7 天的数据
    ul = data.find('ul')                        # 找到所有的 ul 标签
    li = ul.find_all('li')                      # 找到左右的 li 标签
    i = 0                   # 控制爬取的天数
    for day in li:                                  # 遍历找到的每一个 li
        if i < 7 and i > 0:
            temp = []                                       # 临时存放每天的数据
            date = day.find('h1').string            # 得到日期
```

```
                date = date[0:date.index('日')]      # 取出日期号
                temp.append(date)
                inf = day.find_all('p')              # 找出 li 下面的 p 标签,提取第一个 p 标签的
值，即天气
                temp.append(inf[0].string)

                tem_low = inf[1].find('i').string    # 找到最低气温

                if inf[1].find('span') is None:      # 天气预报可能没有最高气温
                    tem_high = None
                else:
                    tem_high = inf[1].find('span').string   # 找到最高气温
                temp.append(tem_low[:-1])
                if tem_high[-1] == '℃':
                    temp.append(tem_high[:-1])
                else:
                    temp.append(tem_high)

                wind = inf[2].find_all('span')       # 找到风向
                for j in wind:
                    temp.append(j['title'])

                wind_scale = inf[2].find('i').string # 找到风级
                index1 = wind_scale.index('级')
                temp.append(int(wind_scale[index1-1:index1]))
                final.append(temp)
            i = i + 1
    return final_day,final
    #print(final)
def get_content2(html):
    """处理得到有用信息保存数据文件"""
    final = []                                       # 初始化一个列
表保存数据
    bs = BeautifulSoup(html, "html.parser")          # 创建 BeautifulSoup
对象
    body = bs.body
    data = body.find('div', {'id': '15d'})           # 找到 div 标签且 id = 15d
    ul = data.find('ul')                             # 找到所有的 ul 标签
    li = ul.find_all('li')                           # 找到左右的 li 标签
    final = []
    i = 0                                            # 控制爬取
的天数
    for day in li:                                   # 遍历找到的每一个
li
```

```
            if i < 8:
                temp = []                                                    # 临时存放
每天的数据
                date = day.find('span',{'class':'time'}).string             # 得到日期
                date = date[date.index('（')+1:-2]                           # 取出日期号
                temp.append(date)
                weather = day.find('span',{'class':'wea'}).string           # 找到天气
                temp.append(weather)
                tem = day.find('span',{'class':'tem'}).text                  # 找到温度
                temp.append(tem[tem.index('/')+1:-1])                       # 找到最低气温
                temp.append(tem[:tem.index('/')-1])                         # 找到最高气温
                wind = day.find('span',{'class':'wind'}).string             # 找到风向
                if '转' in wind:                                             # 如果有风向变
化
                    temp.append(wind[:wind.index('转')])
                    temp.append(wind[wind.index('转')+1:])
                else:                                                       # 如果没有风向
变化，前后风向一致
                    temp.append(wind)
                    temp.append(wind)
                wind_scale = day.find('span',{'class':'wind1'}).string      # 找到风级
                index1 = wind_scale.index('级')
                temp.append(int(wind_scale[index1-1:index1]))

                final.append(temp)
    return final

def write_to_csv(file_name, data, day=14):
    """保存为 csv 文件"""
    with open(file_name, 'a', errors='ignore', newline='') as f:
        if day == 14:
            header = ['日期','天气','最低气温','最高气温','风向 1','风向 2','风级']
        else:
            header = ['小时','温度','风力方向','风级','降水量','相对湿度','空气质量']
        f_csv = csv.writer(f)
        f_csv.writerow(header)
        f_csv.writerows(data)

def main():
    """主函数"""
    print("Weather test")
    # 金昌
    url1 = 'http://www.weather.com.cn/weather/101160601.shtml'      # 7 天天气中国天气网
    url2 = 'http://www.weather.com.cn/weather15d/101160601.shtml' # 8-15 天天气中国天气网
```

```
    html1 = getHTMLtext(url1)
    data1, data1_7 = get_content(html1)          # 获得 1-7 天和当天的数据

    html2 = getHTMLtext(url2)
    data8_14 = get_content2(html2)               # 获得 8-14 天数据
    data14 = data1_7 + data8_14
    #print(data)
    write_to_csv('weather14.csv',data14,14)      # 保存为 csv 文件
    write_to_csv('weather1.csv',data1,1)

if __name__ == '__main__':
    main()


# data14_analysis.py

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import math

def tem_curve(data):
    """温度曲线绘制"""
    date = list(data['日期'])
    tem_low = list(data['最低气温'])
    tem_high = list(data['最高气温'])
    for i in range(0, 14):
        if math.isnan(tem_low[i]) == True:
            tem_low[i] = tem_low[i - 1]
        if math.isnan(tem_high[i]) == True:
            tem_high[i] = tem_high[i - 1]

    tem_high_ave = sum(tem_high) / 14   # 求平均高温
    tem_low_ave = sum(tem_low) / 14   # 求平均低温

    tem_max = max(tem_high)
    tem_max_date = tem_high.index(tem_max)   # 求最高温度
    tem_min = min(tem_low)
    tem_min_date = tem_low.index(tem_min)   # 求最低温度

    x = range(1, 15)
    plt.figure(1)
    plt.plot(x, tem_high, color='red', label='高温')   # 画出高温度曲线
```

```
    plt.scatter(x, tem_high, color='red')  # 点出每个时刻的温度点
    plt.plot(x, tem_low, color='blue', label='低温')  # 画出低温度曲线
    plt.scatter(x, tem_low, color='blue')  # 点出每个时刻的温度点

    plt.plot([1, 15], [tem_high_ave, tem_high_ave], c='black', linestyle='--')  # 画出平均温度虚线
    plt.plot([1, 15], [tem_low_ave, tem_low_ave], c='black', linestyle='--')  # 画出平均温度虚线
    plt.legend()
    plt.text(tem_max_date + 0.15, tem_max + 0.15, str(tem_max), ha='center', va='bottom', fontsize=10.5)  # 标出最高温度
    plt.text(tem_min_date + 0.15, tem_min + 0.15, str(tem_min), ha='center', va='bottom', fontsize=10.5)  # 标出最低温度
    plt.xticks(x)
    plt.title('未来 14 天高温低温变化曲线图')
    plt.xlabel('未来天数/天')
    plt.ylabel('摄氏度/℃')
    plt.show()

def change_wind(wind):
    """改变风向"""
    x = ["北风","南风","西风","东风","东北风","西北风","西南风","东南风"]
    y = [90,270,180,360,45,135,225,315]
    for i in range(0,14):
        for j in range(len(x)):
            if wind[i] == x[j]:
                wind[i] = y[j]
    return wind

def wind_radar(data):
    """风向雷达图"""
    wind1 = list(data['风向 1'])
    wind2 = list(data['风向 2'])
    wind_speed = list(data['风级'])
    wind1 = change_wind(wind1)
    wind2 = change_wind(wind2)

    degs = np.arange(45, 361, 45)
    temp = []
    for deg in degs:
        speed = []
        # 获取 wind_deg 在指定范围的风速平均值数据
        for i in range(0, 14):
            if wind1[i] == deg:
```

```
                    speed.append(wind_speed[i])
                if wind2[i] == deg:
                    speed.append(wind_speed[i])
            if len(speed) == 0:
                temp.append(0)
            else:
                temp.append(sum(speed) / len(speed))
        print(temp)
        N = 8
        theta = np.arange(0. + np.pi / 8, 2 * np.pi + np.pi / 8, 2 * np.pi / 8)
        # 数据极径
        radii = np.array(temp)
        # 绘制极区图坐标系
        plt.axes(polar=True)
        # 定义每个扇区的 RGB 值（R,G,B），x 越大，对应的颜色越接近蓝色
        colors = [(1 - x / max(temp), 1 - x / max(temp), 0.6) for x in radii]
        plt.bar(theta, radii, width=(2 * np.pi / N), bottom=0.0, color=colors)
        plt.title('未来 14 天风级图', x=0.2, fontsize=20)
        plt.show()

def weather_pie(data):
        """绘制天气饼图"""
        weather = list(data['天气'])
        dic_wea = {}
        for i in range(0, 14):
            if weather[i] in dic_wea.keys():
                dic_wea[weather[i]] += 1
            else:
                dic_wea[weather[i]] = 1
        print(dic_wea)
        explode = [0.01] * len(dic_wea.keys())
        color = ['lightskyblue', 'silver', 'yellow', 'salmon', 'grey', 'lime', 'gold', 'red', 'green', 'pink']
        plt.pie(dic_wea.values(), explode=explode, labels=dic_wea.keys(), autopct='%1.1f%%',
colors=color)
        plt.title('未来 14 天气候分布饼图')
        plt.show()

def main():
        plt.rcParams['font.sans-serif'] = ['SimHei']  # 解决中文显示问题
        plt.rcParams['axes.unicode_minus'] = False  # 解决负号显示问题
        data14 = pd.read_csv('weather14.csv', encoding='gb2312')
        # print(data14)
        tem_curve(data14)#未来 14 天高低温度变化
        wind_radar(data14)#未来 14 天风级图
```

```
    weather_pie(data14)#未来 14 天气候分布饼图

if __name__ == '__main__':
    main()
```

4.南昌 14 天

```
# -*- coding: utf-8 -*-
"""
Created on Sun Jul  4 08:51:31 2021

@author: 29379
"""
# weather.py
import requests
from bs4 import BeautifulSoup
import csv
import json
def getHTMLtext(url):
    """请求获得网页内容"""
    try:
        r = requests.get(url, timeout = 30)
        r.raise_for_status()
        r.encoding = r.apparent_encoding
        print("成功访问")
        return r.text
    except:
        print("访问错误")
        return" "

def get_content(html):
    """处理得到有用信息保存数据文件"""
    final = []                                  # 初始化一个列表保存数据
    bs = BeautifulSoup(html, "html.parser")   # 创建 BeautifulSoup 对象
    body = bs.body
    data = body.find('div', {'id': '7d'})       # 找到 div 标签且 id = 7d
    # 下面爬取当天的数据
    data2 = body.find_all('div',{'class':'left-div'})
    text = data2[2].find('script').string
    text = text[text.index('=')+1 :-2]          # 移除改 var data=将其变为 json 数据
    jd = json.loads(text)
    dayone = jd['od']['od2']                    # 找到当天的数据
    final_day = []                                  # 存放当天的数据
    count = 0
```

```
    for i in dayone:
        temp = []
        if count <=23:
            temp.append(i['od21'])                  # 添加时间
            temp.append(i['od22'])                  # 添加当前时刻温度
            temp.append(i['od24'])                  # 添加当前时刻风力方向
            temp.append(i['od25'])                  # 添加当前时刻风级
            temp.append(i['od26'])                  # 添加当前时刻降水量
            temp.append(i['od27'])                  # 添加当前时刻相对湿度
            temp.append(i['od28'])                  # 添加当前时刻控制质量
            #print(temp)
            final_day.append(temp)
        count = count +1
    # 下面爬取 7 天的数据
    ul = data.find('ul')                        # 找到所有的 ul 标签
    li = ul.find_all('li')                      # 找到左右的 li 标签
    i = 0                   # 控制爬取的天数
    for day in li:                                  # 遍历找到的每一个 li
        if i < 7 and i > 0:
            temp = []                                       # 临时存放每天的数据
            date = day.find('h1').string            # 得到日期
            date = date[0:date.index('日')]         # 取出日期号
            temp.append(date)
            inf = day.find_all('p')                 # 找出 li 下面的 p 标签,提取第一个 p 标签的
值，即天气
            temp.append(inf[0].string)

            tem_low = inf[1].find('i').string       # 找到最低气温

            if inf[1].find('span') is None:         # 天气预报可能没有最高气温
                tem_high = None
            else:
                tem_high = inf[1].find('span').string   # 找到最高气温
            temp.append(tem_low[:-1])
            if tem_high[-1] == '℃':
                temp.append(tem_high[:-1])
            else:
                temp.append(tem_high)

            wind = inf[2].find_all('span')          # 找到风向
            for j in wind:
                temp.append(j['title'])

            wind_scale = inf[2].find('i').string #  找到风级
```

```
                index1 = wind_scale.index('级')
                temp.append(int(wind_scale[index1-1:index1]))
                final.append(temp)
            i = i + 1
    return final_day,final
    #print(final)
def get_content2(html):
    """处理得到有用信息保存数据文件"""
    final = []                                          # 初始化一个列
表保存数据
    bs = BeautifulSoup(html, "html.parser")             # 创建 BeautifulSoup
对象
    body = bs.body
    data = body.find('div', {'id': '15d'})              # 找到 div 标签且 id = 15d
    ul = data.find('ul')                                # 找到所有的 ul 标签
    li = ul.find_all('li')                              # 找到左右的 li 标签
    final = []
    i = 0                                               # 控制爬取
的天数
    for day in li:                                      # 遍历找到的每一个
li
        if i < 8:
            temp = []                                   # 临时存放
每天的数据
            date = day.find('span',{'class':'time'}).string     # 得到日期
            date = date[date.index('（')+1:-2]                  # 取出日期号
            temp.append(date)
            weather = day.find('span',{'class':'wea'}).string   # 找到天气
            temp.append(weather)
            tem = day.find('span',{'class':'tem'}).text         # 找到温度
            temp.append(tem[tem.index('/')+1:-1])               # 找到最低气温
            temp.append(tem[:tem.index('/')-1])                 # 找到最高气温
            wind = day.find('span',{'class':'wind'}).string     # 找到风向
            if '转' in wind:                                    # 如果有风向变
化
                temp.append(wind[:wind.index('转')])
                temp.append(wind[wind.index('转')+1:])
            else:                                       # 如果没有风向
变化，前后风向一致
                temp.append(wind)
                temp.append(wind)
            wind_scale = day.find('span',{'class':'wind1'}).string  # 找到风级
            index1 = wind_scale.index('级')
            temp.append(int(wind_scale[index1-1:index1]))
```

```
                final.append(temp)
    return final

def write_to_csv(file_name, data, day=14):
    """保存为 csv 文件"""
    with open(file_name, 'a', errors='ignore', newline='') as f:
        if day == 14:
            header = ['日期','天气','最低气温','最高气温','风向 1','风向 2','风级']
        else:
            header = ['小时','温度','风力方向','风级','降水量','相对湿度','空气质量']
        f_csv = csv.writer(f)
        f_csv.writerow(header)
        f_csv.writerows(data)

def main():
    """主函数"""
    print("Weather test")
    # 南昌
    url1 = 'http://www.weather.com.cn/weather/101240101.shtml'      # 7 天天气中国天气网
    url2 = 'http://www.weather.com.cn/weather15d/101240101.shtml' # 8-15 天天气中国天气网

    html1 = getHTMLtext(url1)
    data1, data1_7 = get_content(html1)          # 获得 1-7 天和当天的数据

    html2 = getHTMLtext(url2)
    data8_14 = get_content2(html2)               # 获得 8-14 天数据
    data14 = data1_7 + data8_14
    #print(data)
    write_to_csv('weather14.csv',data14,14)      # 保存为 csv 文件
    write_to_csv('weather1.csv',data1,1)

if __name__ == '__main__':
    main()


# data14_analysis.py

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import math

def tem_curve(data):
```

```
    """温度曲线绘制"""
    date = list(data['日期'])
    tem_low = list(data['最低气温'])
    tem_high = list(data['最高气温'])
    for i in range(0, 14):
        if math.isnan(tem_low[i]) == True:
            tem_low[i] = tem_low[i - 1]
        if math.isnan(tem_high[i]) == True:
            tem_high[i] = tem_high[i - 1]

    tem_high_ave = sum(tem_high) / 14  # 求平均高温
    tem_low_ave = sum(tem_low) / 14  # 求平均低温

    tem_max = max(tem_high)
    tem_max_date = tem_high.index(tem_max)  # 求最高温度
    tem_min = min(tem_low)
    tem_min_date = tem_low.index(tem_min)  # 求最低温度

    x = range(1, 15)
    plt.figure(1)
    plt.plot(x, tem_high, color='red', label='高温')  # 画出高温度曲线
    plt.scatter(x, tem_high, color='red')  # 点出每个时刻的温度点
    plt.plot(x, tem_low, color='blue', label='低温')  # 画出低温度曲线
    plt.scatter(x, tem_low, color='blue')  # 点出每个时刻的温度点

    plt.plot([1, 15], [tem_high_ave, tem_high_ave], c='black', linestyle='--')  # 画出平均温度虚
线
    plt.plot([1, 15], [tem_low_ave, tem_low_ave], c='black', linestyle='--')  # 画出平均温度虚
线
    plt.legend()
    plt.text(tem_max_date + 0.15, tem_max + 0.15, str(tem_max), ha='center', va='bottom',
fontsize=10.5)  # 标出最高温度
    plt.text(tem_min_date + 0.15, tem_min + 0.15, str(tem_min), ha='center', va='bottom',
fontsize=10.5)  # 标出最低温度
    plt.xticks(x)
    plt.title('未来 14 天高温低温变化曲线图')
    plt.xlabel('未来天数/天')
    plt.ylabel('摄氏度/℃')
    plt.show()

def change_wind(wind):
    """改变风向"""
    x = ["北风","南风","西风","东风","东北风","西北风","西南风","东南风"]
    y = [90,270,180,360,45,135,225,315]
```

```
    for i in range(0,14):
        for j in range(len(x)):
            if wind[i] == x[j]:
                wind[i] = y[j]
    return wind

def wind_radar(data):
    """风向雷达图"""
    wind1 = list(data['风向 1'])
    wind2 = list(data['风向 2'])
    wind_speed = list(data['风级'])
    wind1 = change_wind(wind1)
    wind2 = change_wind(wind2)

    degs = np.arange(45, 361, 45)
    temp = []
    for deg in degs:
        speed = []
        # 获取 wind_deg 在指定范围的风速平均值数据
        for i in range(0, 14):
            if wind1[i] == deg:
                speed.append(wind_speed[i])
            if wind2[i] == deg:
                speed.append(wind_speed[i])
        if len(speed) == 0:
            temp.append(0)
        else:
            temp.append(sum(speed) / len(speed))
    print(temp)
    N = 8
    theta = np.arange(0. + np.pi / 8, 2 * np.pi + np.pi / 8, 2 * np.pi / 8)
    # 数据极径
    radii = np.array(temp)
    # 绘制极区图坐标系
    plt.axes(polar=True)
    # 定义每个扇区的 RGB 值（R,G,B），x 越大，对应的颜色越接近蓝色
    colors = [(1 - x / max(temp), 1 - x / max(temp), 0.6) for x in radii]
    plt.bar(theta, radii, width=(2 * np.pi / N), bottom=0.0, color=colors)
    plt.title('未来 14 天风级图', x=0.2, fontsize=20)
    plt.show()

def weather_pie(data):
    """绘制天气饼图"""
    weather = list(data['天气'])
```

```
    dic_wea = {}
    for i in range(0, 14):
        if weather[i] in dic_wea.keys():
            dic_wea[weather[i]] += 1
        else:
            dic_wea[weather[i]] = 1
    print(dic_wea)
    explode = [0.01] * len(dic_wea.keys())
    color = ['lightskyblue', 'silver', 'yellow', 'salmon', 'grey', 'lime', 'gold', 'red', 'green', 'pink']
    plt.pie(dic_wea.values(), explode=explode, labels=dic_wea.keys(), autopct='%1.1f%%',
colors=color)
    plt.title('未来 14 天气候分布饼图')
    plt.show()

def main():
    plt.rcParams['font.sans-serif'] = ['SimHei']  # 解决中文显示问题
    plt.rcParams['axes.unicode_minus'] = False  # 解决负号显示问题
    data14 = pd.read_csv('weather14.csv', encoding='gb2312')
    # print(data14)
    tem_curve(data14)#未来 14 天高低温度变化
    wind_radar(data14)#未来 14 天风级图
    weather_pie(data14)#未来 14 天气候分布饼图

if __name__ == '__main__':
    main()
```