

期末大作业-五个城市的租房数据分析实验报告

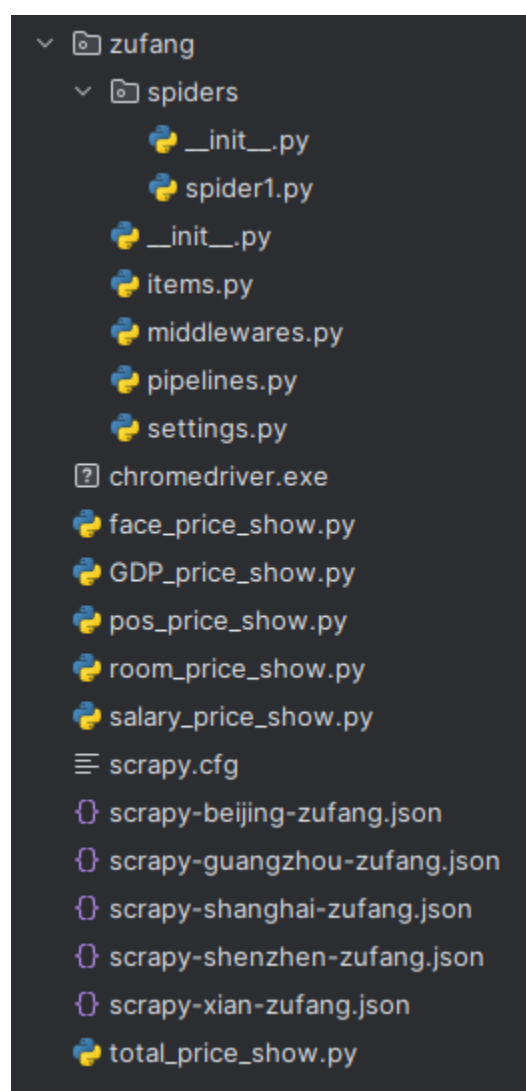
学号：2021211338 姓名：郭柏彤 班级：310

一、总体设计：

本次租房数据分析大作业主要包括三大部分，第一部分为爬虫文件，包括了对五个城市（北上广深和西安）数据的爬取操作。第二部分为 json 文件，包括爬取下来的五个城市的租房具体数据。第三部分为数据处理和可视化展示部分。

其中，第三部分为完全自主设计，第一部分则在 scrapy 爬虫框架的基础上进行修改设计。

总体文件结构为：



1.爬虫部分设计：

在 scrapy 框架的基础上进行设计。其中，包含了 spider1.py，为主要的爬虫爬取分析文件，包括了爬取网址的设置和 xpath 路径解析，items.py 为爬取对象参数的设置，middlewares.py 为框架自带代码，

pipelines.py 为管道的设置，在这里设置爬取下来的数据存储的文件。
Settings.py 为爬虫的一些参数设置，在这里进行了部分参数的修改。

2.存储部分设计

分为五个文件进行存储,

scrapy-beijing-zufang.json: 北京的租房信息数据存儲

scrapy-quangzhou-zufang.json: 广州的租房信息数据存储

scrapy-shanghai-zufang.json: 上海的租房信息数据存储

scrapy-shenzhen-zufang.json: 深圳的租房信息数据存储

scrapy-xian-zufang.json: 西安的租房信息数据存储

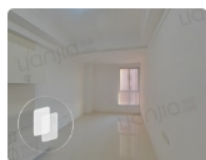
其中，通过链家官网可知，每个城市的租房数据共 100 页，每页 30 条，因此每个城市的数据有 3000 条，五个城市的数据共 15000 条。

Json 文件的保存的一条租房数据如下所示:

```
{ "title": "整租·六铺炕一区 2 室 1 厅 南", "price": "7000", "position0": "西城", "position1": "六铺炕", "position2": "六铺炕一区", "information": "西城-六铺炕-六铺炕一区 / 52.00 m² / 南 / 2 室 1 厅 1 卫 / 高楼层 (5 层)" }
```

{"title": "碧桂园·六福城一期 2室1厅 南", "price": "7000", "position0": "西瓠", "position1": "六福城", "position2": "六福城一期", "information": "西瓠·六福城·六福城一期 / 52.00\u00b0 / \u20132999 \u00b0", "type": "二手房"},
 {"title": "碧桂园·珑悦城二期 2室1厅 南", "price": "5990", "position0": "朝阳", "position1": "珑悦城", "position2": "珑悦城二期", "information": "珑悦城·朝阳·珑悦城二期 / 52.29\u00b0 / 西瓠 / 3", "type": "二手房"},
 {"title": "碧桂园·长阳半岛三期 3室1厅 南", "price": "4700", "position0": "房山", "position1": "长阳", "position2": "长阳半岛三期", "information": "房山·长阳·长阳半岛三期 / 99.59\u00b0 / 西瓠 / 3", "type": "二手房"},
 {"title": "绿城·京悦府 北四期独栋 舒朗庭院双卫+三露台 1室1厅", "price": "1752-1924", "position0": "null", "position1": "null", "position2": "null", "information": "京悦府/长 / 14.00-15.00\u00b0", "type": "二手房"},
 {"title": "碧桂园·京悦府三期 3室1厅 南北", "price": "5000", "position0": "丰台", "position1": "方庄", "position2": "京悦府三期", "information": "丰台·方庄·京悦府三期 / 47.82\u00b0 / 西瓠 / 1", "type": "二手房"},
 {"title": "绿城·康桥创意公园 朝晖公寓 高马路 南苑 阳光大床房 开间", "price": "5900-6100", "position0": "null", "position1": "null", "position2": "null", "information": "京悦府/长 / 25.00\u00b0 / 3", "type": "二手房"},
 {"title": "碧桂园·万寿宫 2室1厅 东", "price": "7700", "position0": "朝阳", "position1": "望京", "position2": "万寿宫", "information": "朝阳·望京园·万寿宫 / 89.00\u00b0 / 东瓠 / 2室1厅1卫 / 3", "type": "二手房"},
 {"title": "绿城·锦悦府 首都图书酒店 光熙里独栋一居室 1室1厅", "price": "7676", "position0": "null", "position1": "null", "position2": "null", "information": "京悦府/长 / 55.00\u00b0 / 12室1厅 / 12", "type": "二手房"},
 {"title": "碧桂园·锦悦府 4室2厅 南", "price": "8500", "position0": "大兴", "position1": "高米店", "position2": "锦悦府", "information": "大兴·高米店·锦悦府 / 147.00\u00b0 / 西瓠 / 4室2厅 / 3", "type": "二手房"},
 {"title": "绿城·柚木里 北京丽泽商务区 现房月租随时看房 嘉达豪府 南苑锦悦站 1室1厅", "price": "2099-2122", "position0": "null", "position1": "null", "position2": "null", "information": "京悦府/长 / 1", "type": "二手房"},
 {"title": "碧桂园·世华光熙二期 2室1厅 南北", "price": "7500", "position0": "海淀", "position1": "青西", "position2": "世华光熙二期", "information": "海淀·青西·世华光熙二期 / 80.00\u00b0 / 3", "type": "二手房"},
 {"title": "绿城·恒景里光熙 星光二角 可开发零租金 1室1厅", "price": "3501", "position0": "null", "position1": "null", "position2": "null", "information": "京悦府/长 / 63.00\u00b0 / 2间在租 / 1室1厅1卫", "type": "二手房"},
 {"title": "碧桂园·望都新城二期 2室1厅 南", "price": "4000", "position0": "昌平", "position1": "北七家", "position2": "望都新城二期", "information": "昌平·北七家·望都新城二期 / 89.00\u00b0 / 3", "type": "二手房"},
 {"title": "绿城·柚木里 北京丽泽商务区 现房4月待价 华立世纪无租金零首付月付 近地铁花梨树 1室1厅", "price": "2746-2881", "position0": "null", "position1": "null", "position2": "null", "information": "京悦府/长 / 1", "type": "二手房"},
 {"title": "碧桂园·望都新城二期 2室1厅 南", "price": "7000", "position0": "昌平", "position1": "东小口", "position2": "望都新城二期", "information": "昌平·东小口·望都新城二期 / 69.90\u00b0 / 南 / 2室1厅1卫 / 3", "type": "二手房"},
 {"title": "碧桂园·MASTER公馆 1室1厅 东北", "price": "9600", "position0": "朝阳", "position1": "国贸", "position2": "MASTER公馆", "information": "朝阳·国贸·MASTER公馆 / 94.21\u00b0 / 东瓠 / 1室", "type": "二手房"},
 {"title": "碧桂园·锦悦府 3室2厅 东", "price": "14000", "position0": "大兴", "position1": "东马厂", "position2": "锦悦府", "information": "大兴·东马厂·锦悦府 / 243.76\u00b0 / 西瓠 / 3室2厅 / 3", "type": "二手房"},
 {"title": "绿城·京悦府公寓 桃红杏雨嘉悦里独栋 阔气轻奢豪庭公寓 开间", "price": "3300-3500", "position0": "null", "position1": "null", "position2": "null", "information": "京悦府/长 / 25.00\u00b0 / 60套在租 / 1室", "type": "二手房"},
 {"title": "碧桂园·华颂城二期 4室 西", "price": "15580", "position0": "朝阳", "position1": "东直门", "position2": "华颂城二期", "information": "朝阳·东直门·华颂城二期 / 130.43\u00b0 / 西瓠 / 1室", "type": "二手房"},
 {"title": "碧桂园·长阳半岛怡和南8号院 3室2厅 南/北", "price": "6700", "position0": "房山", "position1": "长阳", "position2": "长阳半岛怡和南8号院", "information": "房山·长阳·长阳半岛怡和南8号院 / 117.14\u00b0 / 南 / 北", "type": "二手房"},
 {"title": "碧桂园·保利中央公园 3室2厅 南", "price": "26000", "position0": "朝阳", "position1": "望京", "position2": "保利中央公园", "information": "朝阳·望京·保利中央公园 / 117.14\u00b0 / 南 / 北", "type": "二手房"},
 {"title": "绿城·锦悦府公寓 京悦府锦悦站 阔气轻奢豪庭公寓无租金 开间", "price": "4850", "position0": "null", "position1": "null", "position2": "null", "information": "京悦府/长 / 20.28-21.14\u00b0 / 10套在租 / 1室", "type": "二手房"}

而链家官网上的一条租房信息如下:



整租·亿润花园 1室0厅 南/北 必看好房

1200元/月

莲湖-城西-亿润花园 / 33.00m² / 南北 / 1室0厅1卫

集中供暖 随时看房

🕒 3天前维护

title 为标题部分;

position1 为位置的第一部分， 图中所示为莲湖

position2 为位置的第二部分， 图中所示为城西

position3 为位置的第三部分， 图中所示为亿润花园

information 为其他信息，包括图中的第二行的所有信息。

3.数据处理与展示部分设计.

该部分包括六个 python 程序，每个程序分别对应作业中的一条要求。

total_price_show.py: 该文件实现了对五个城市的租房价格的均价, 中位数, 最高价, 最低价和单位面积的均价, 中位数, 最高价, 最低价的比较分析和图表绘制

room_price_show.py: 该文件实现了对五个城市按照居室展示, 几居即几室, 分别展示了 1 居室, 2 居室, 3 居室的均价, 中位数, 最高价, 最低价的比较分析和图表绘制

pos_price_show.py: 该文件实现了按照板块来展示不同城市不同板块的均价的图表绘制
face_price_show.py: 该文件实现了对五个不同城市按照房屋面向来进行均价的比较分析和图表绘制
GDP_price_show.py: 该文件实现了对五个城市租房均价和 GDP 的比较分析和图表绘制
salary_price_show.py: 该文件实现了对五个城市租房均价和工资水平的比较分析和图表绘制

二、具体实现过程

第一部分：数据的爬取

本次爬取的目标网址为链家官网的租房页面。选取的五个城市分别为北京，上海，广州，深圳，和自主选取的一个城市西安。

爬取的五个网址分别为：

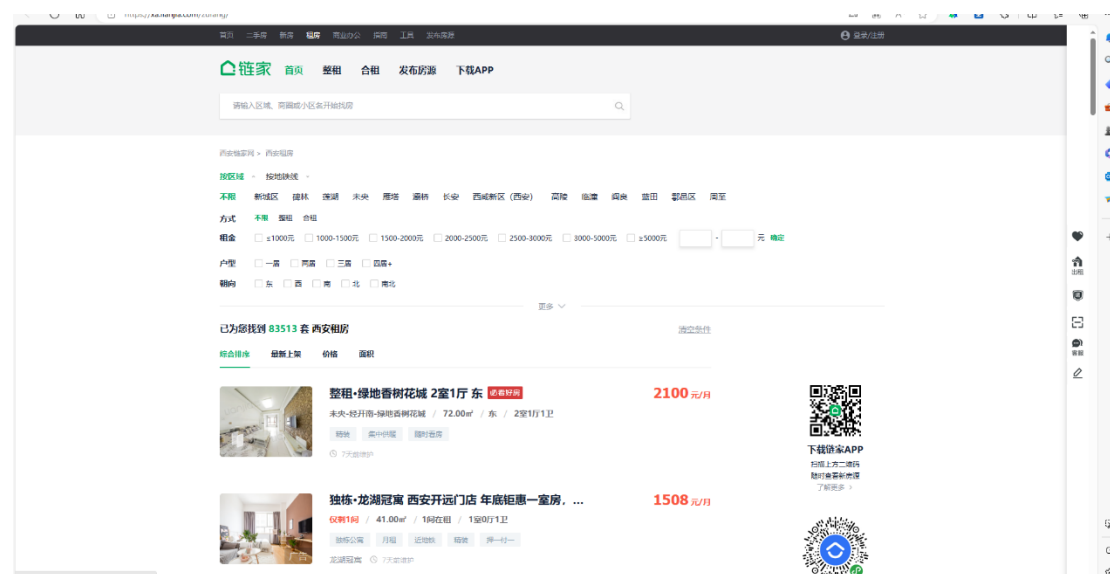
<https://bj.lianjia.com/zufang/>

<https://sh.lianjia.com/zufang/>

<https://gz.lianjia.com/zufang/>

<https://sz.lianjia.com/zufang/>

<https://xa.lianjia.com/zufang/>



定义一个名为各个城市名称拼音的爬虫，设置好基础的起始页面等信息：

```
class Zufangspider(scrapy.spiders.Spider):
```

```
    name = "xian" # 爬虫名字分别为 beijing shanghai guangzhou shenzhen xian
    allowed_domains = ["xa.lianjia.com"] # 爬取的起始页面
    start_urls = []
    custom_settings = {
        'ITEM_PIPELINES': {'zufang.pipelines.zufangline': 300},
    }
```

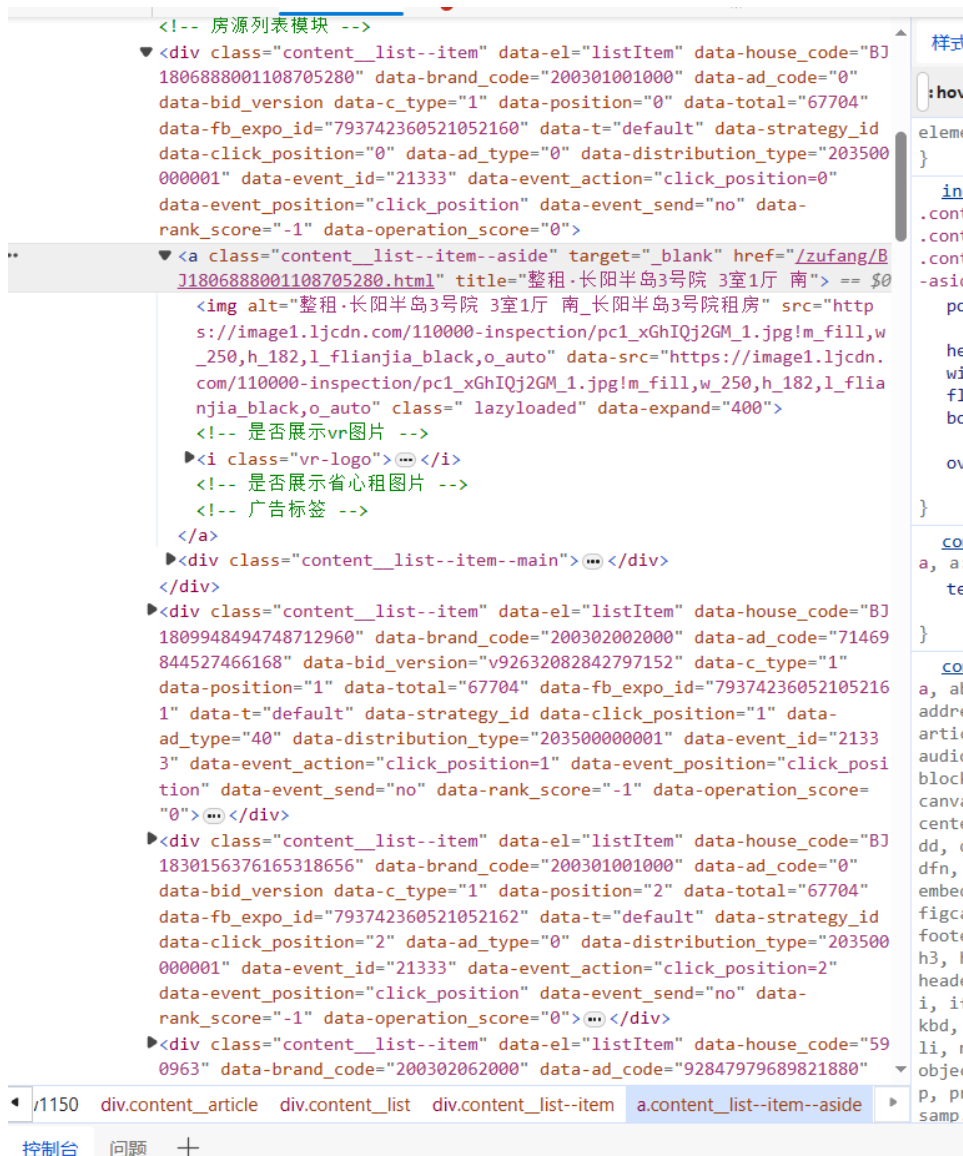
通过观察链家官网页面，可以发现每个城市的租房信息都一共有 100 页：



因此，可以通过遍历 1-100 来很方便的设置要爬取的 100 个页面。
python 源代码如下：

```
for page in range(1, 101): # 共 100 页，所以利用一个循环来爬取
    url1 = 'https://xa.lianjia.com/zufang/pg{}/'.format(page)
    start_urls.append(url1)
```

在设置好需要爬取的页面后，开始对 xpath 路径进行分析获取需要的信息。
在浏览器中右键检查网游的源代码：



此时，便可以看到一条条的租房信息对应的 div，右键要保留的信息即可复制 xpath 路径。
在网页中，我们需要保存的信息有一条租房信息的标题，价格，地点，以及其他信息。
设置好 xpath 路径进行存储即可。

python 源代码：

```
def parse(self, response, **kwargs):

    item = zufangitem()
    div_list = response.xpath("//*[@id=\"content\"]/div[1]/div[1]/div")

    # 通过 XPATH 来分析爬取到的内容，并提取需要的数据
    for each in div_list:
        item['title'] = each.xpath("normalize-space(./div/p[1]/a/text())").extract_first()
        item['price'] = each.xpath("normalize-space(./div/span/em/text())").extract_first()
        item['position0'] = each.xpath("./div/p[2]/a[1]/text()").extract_first()
        item['position1'] = each.xpath("./div/p[2]/a[2]/text()").extract_first()
        item['position2'] = each.xpath("./div/p[2]/a[3]/text()").extract_first()
        item['information'] = each.xpath("normalize-space(./div/p[2])").extract_first()
    yield item
```

对于需要保存的对象，在 item 文件中列出：

```
class zufangitem(scrapy.Item):
    title = scrapy.Field() # 标题
    price = scrapy.Field() # 月租金
    position0 = scrapy.Field() # 地址 1
    position1 = scrapy.Field() # 地址 2
    position2 = scrapy.Field() # 地址 3
    information = scrapy.Field() # 其他信息
```

同时，修改管道文件，设置好爬取下来的数据要保存的文件名：

```
class zufangline(object):
    def open_spider(self, spider):
        try:
            self.file = open('scrapy-xian-zufang.json', 'w', encoding="utf-8")
        except Exception as err:
            print(err)

    def process_item(self, item, spider):
        dict_item = dict(item)
        json_str = json.dumps(dict_item, ensure_ascii=False) + "\n"
        self.file.write(json_str)
        return item

    def close_spider(self, spider):
        self.file.close()
```

其中，self.file = open('scrapy-xian-zufang.json', 'w', encoding="utf-8")即为设置保存的文件名的部分。

最后，修改爬虫 setting 里的部分设置，如加上随机的 user_agent 包头等，来避免网站的自动爬虫检测。

```
BOT_NAME = "zufang"
#2403:a200:a200:13f1:183:84:18:11

SPIDER_MODULES = ["zufang.spiders"]
NEWSPIDER_MODULE = "zufang.spiders"

# Crawl responsibly by identifying yourself (and your website) on the user-agent
#USER_AGENT = "python_gm (+http://www.yourdomain.com)"
#USER_AGENT = "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.77 Safari/537.36"
#USER_AGENT = "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36 Edg/119.0.0.0"
DOWNLOADER_MIDDLEWARES = {
    # 'lianjia.middlewares.LianjiaDownloaderMiddleware': 543,
    'zufang.middlewares.RandomUserAgentMiddleware': 900,
}

MY_USER_AGENT = [
    "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; AcooBrowser; .NET CLR 1.1.4322; .NET CLR 2.0.50727)",
    "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; Acoo Browser; SLCC1; .NET CLR 2.0.50727; Media Center PC 5.0; .NET CLR 3.0.04506)",
    "Mozilla/4.0 (compatible; MSIE 7.0; AOL 9.5; AOLBuild 4337.35; Windows NT 5.1; .NET CLR 1.1.4322; .NET CLR 2.0.50727)",
    "Mozilla/5.0 (Windows; U; MSIE 9.0; Windows NT 9.0; en-US)",
    "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0; .NET CLR 3.5.30729; .NET CLR 3.0.30729; .NET CLR 2.0.50727; Media Center PC 6.0)",
    "Mozilla/5.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; .NET CLR 1.0.3705; .NET CLR 3.0.04506.30)",
    "Mozilla/4.0 (compatible; MSIE 7.0b; Windows NT 5.2; .NET CLR 1.1.4322; .NET CLR 2.0.50727; InfoPath.2; .NET CLR 3.0.04506.30)",
    "Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-CN) AppleWebKit/525.15 (KHTML, like Gecko, Safari/419.3) Arora/0.3 (Change: 287 c9dfeb30)",
    "Mozilla/5.0 (X11; U; Linux; en-US) AppleWebKit/527+ (KHTML, like Gecko, Safari/419.3) Arora/0.6",
    "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.0.1.2pre) Gecko/20070215 K-MinJae/2.1.1",
    "Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-CN; rv:1.9) Gecko/20080705 Firefox/3.0 Kapiko/3.0",
    "Mozilla/5.0 (X11; Linux i686; U; Gecko/20070322 Kazehakase/0.4.5",
    "Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.0.8) Gecko Fedora/1.9.0.8-1.fc10 Kazehakase/0.5.6",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11 (KHTML, like Gecko) Chrome/17.0.963.56 Safari/535.11",
    "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_3) AppleWebKit/535.20 (KHTML, like Gecko) Chrome/19.0.1036.7 Safari/535.20",
    "Opera/9.80 (Macintosh; Intel Mac OS X 10.6.8; U; fr) Presto/2.9.168 Version/11.52",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.11 (KHTML, like Gecko) Chrome/20.0.1132.11 TaoBrowser/2.0 Safari/536.11",
]
```

```
# Enable or disable downloader middlewares
# See https://docs.scrapy.org/en/latest/topics/downloader-middleware.html
DOWNLOADER_MIDDLEWARES = {
    "zufang.middlewares.ZufangDownloaderMiddleware": 543,
}

# Enable or disable extensions
# See https://docs.scrapy.org/en/latest/topics/extensions.html
EXTENSIONS = {
    # "scrapy.extensions.telnet.TelnetConsole": None,
}

# Configure item pipelines
# See https://docs.scrapy.org/en/latest/topics/item-pipeline.html
ITEM_PIPELINES = {'zufang.pipelines.zufangline': 300, }
#ITEM_PIPELINES = {'lianjia.pipelines.firsthandline': 300}
# Enable and configure the AutoThrottle extension (disabled by default)
# See https://docs.scrapy.org/en/latest/topics/autothrottle.html
```

在上述代码均构建完成后，完成并实现了一个爬虫，在终端控制台中输入指令：
scrapy crawl xian 来运行爬虫，爬取西安的租房数据。其他四个城市同理。

```
PS E:\Python\pachong\zufang> scrapy crawl xian






DevTools listening on ws://127.0.0.1:52767/devtools/browser/21d6cbc0-f65a-4044-89e0-da02fdade1e6
```

运行爬虫，等待运行结束后可以发现 json 文件已经被写入需要的数据。


```
16853 {'title': '高租·翔凤 5居室 西南向', 'price': '800', 'position0': '雁塔', 'position1': '城东', 'position2': '翔凤', 'information': '精选 / 雁塔-城东-翔凤 / 20.00㎡ / 西南 / 5室0厅 2999 / 4  
16854 {'title': '整租·华润置地丰城市DK3 3室2厅 南/北', 'price': '3400', 'position0': '雁塔', 'position1': '国际港务区', 'position2': '华润置地丰城市DK3', 'information': '雁塔-国际港务区-华润置地  
16855 {'title': '整租·海璟珑樾的舍 3室2厅 东/南', 'price': '2700', 'position0': '未央', 'position1': '经开南', 'position2': '海璟珑樾的舍', 'information': '未央-经开南-海璟珑樾的舍 / 93.00㎡ / 东  
16856 {'title': '整租·绿城公园首府 2室1厅 南', 'price': '2500', 'position0': '莲湖', 'position1': '城西', 'position2': '绿城公园首府', 'information': '莲湖-城西-绿城公园首府 / 89.00㎡ / 南 / 2室1厅  
16857 {'title': '整租·中海凯旋门 4室2厅 南/北', 'price': '5000', 'position0': '雁塔', 'position1': '曲江', 'position2': '中海凯旋门', 'information': '雁塔-曲江-中海凯旋门 / 143.00㎡ / 南 北 / 4室2厅  
16858 {'title': '整租·新天地小区 3居室 南卧', 'price': '550', 'position0': '雁塔', 'position1': '城南', 'position2': '新天地小区', 'information': '精选 / 雁塔-城南-新天地小区 / 18.00㎡ / 南 / 3室2厅  
16859 {'title': '整租·慈云宏景 3室2厅 南/北', 'price': '2600', 'position0': '未央', 'position1': '经开北', 'position2': '慈云宏景', 'information': '未央-经开北-慈云宏景 / 120.00㎡ / 南 北 / 3室2厅  
16860 {'title': '整租·紫薇田园都市A区 5居室 南卧', 'price': '920', 'position0': '长安', 'position1': '紫薇田园都市', 'position2': '紫薇田园都市A区', 'information': '精选 / 长安-紫薇田园都市-紫薇田园  
16861 {'title': '整租·枫叶苑南区 2室1厅 南/北', 'price': '2400', 'position0': '雁塔', 'position1': '高新一中', 'position2': '枫叶苑南区', 'information': '雁塔-高新一中-枫叶苑南区 / 83.07㎡ / 南 北  
16862 {'title': '整租·中华世纪城 4室2厅 南/北', 'price': '33000', 'position0': '雁塔', 'position1': '唐延路', 'position2': '中华世纪城', 'information': '精选 / 雁塔-唐延路-中华世纪城 / 242.74㎡ /  
16863 {'title': '整租·卡布奇诺公馆 3居室 南卧', 'price': '1000', 'position0': '雁塔', 'position1': '高新六路', 'position2': '卡布奇诺公馆', 'information': '精选 / 雁塔-高新六路-卡布奇诺公馆 / 24.00㎡ /  
16864 {'title': '整租·高阳湖九都 2室2厅 南', 'price': '1600', 'position0': '高陵', 'position1': '高陵', 'position2': '高阳湖九都', 'information': '高陵-高陵-高阳湖九都 / 88.00㎡ / 南 / 2室2厅  
16865 {'title': '整租·宏府鹏瑞九天 4居室 南卧', 'price': '700', 'position0': '莲湖', 'position1': '城西', 'position2': '宏府鹏瑞九天', 'information': '精选 / 莲湖-城西-宏府鹏瑞九天 / 26.00㎡ / 南 /  
16866 {'title': '整租·东南悦湖 1室0厅 南', 'price': '1600', 'position0': '莲湖', 'position1': '城西', 'position2': '东南悦湖', 'information': '莲湖-城西-东南悦湖 / 50.00㎡ / 南 / 1室0厅1卫 / 高阳湖  
16867 {'title': '整租·山水怡景南区 2室2厅 南', 'price': '2400', 'position0': '长安', 'position1': '长安', 'position2': '山水怡景南区', 'information': '精选 / 长安-长安-山水怡景南区 / 90.00㎡ / 南 / 2室2厅  
16868 {'title': '整租·民航社区 4居室 南卧', 'price': '550', 'position0': '莲湖', 'position1': '城西', 'position2': '民航社区', 'information': '精选 / 莲湖-城西-民航社区 / 10.00㎡ / 南 / 4室0厅1卫  
16869 {'title': '整租·当代宏府MOMA 3室2厅 西', 'price': '2800', 'position0': '莲湖', 'position1': '城西', 'position2': '当代宏府MOMA', 'information': '精选 / 莲湖-城西-当代宏府MOMA / 98.00㎡ / 西 / 3室  
1700 {'title': '整租·昆明大厦 4居室 南卧', 'price': '550', 'position0': '莲湖', 'position1': '城西', 'position2': '昆明大厦', 'information': '精选 / 莲湖-城西-昆明大厦 / 12.00㎡ / 南 / 4室1厅2卫  
1701 {'title': '整租·翡翠岭 4室2厅 南/北', 'price': '9500', 'position0': '雁塔', 'position1': '曲江', 'position2': '翡翠岭', 'information': '精选 / 雁塔-曲江-翡翠岭 / 166.42㎡ / 南 北 / 4室  
1702 {'title': '整租·中庭城市花园 4居室 南卧', 'price': '953', 'position0': '未央', 'position1': '经开南', 'position2': '中庭城市花园', 'information': '精选 / 未央-经开南-中庭城市花园 / 23.00㎡ /  
1703 {'title': '整租·西安高新锦汇社区0区 1室1厅 南', 'price': '2100', 'position0': '长安', 'position1': '北正街', 'position2': '西安高新锦汇社区0区', 'information': '精选 / 长安-北正街-西安高新锦汇社区  
1704 {'title': '整租·海璟新天地 2室1厅 东南', 'price': '2400', 'position0': '未央', 'position1': '经开南', 'position2': '海璟新天地', 'information': '未央-经开南-海璟新天地 / 105.00㎡ / 东南 / 2室  
1705 {'title': '整租·山水怡景南区 3室2厅 南', 'price': '2300', 'position0': '长安', 'position1': '长安', 'position2': '山水怡景南区', 'information': '长安-长安-山水怡景南区 / 118.00㎡ / 南 / 3室2厅  
1706 {'title': '整租·新城 5居室 南卧', 'price': '800', 'position0': '雁塔', 'position1': '高新六路', 'position2': '新城', 'information': '雁塔-高新六路-新城 / 26.00㎡ / 南 / 5室0厅2卫 / 中庭  
1707 {'title': '整租·千禧社区 2室1厅 南', 'price': '1800', 'position0': '未央', 'position1': '北窑站', 'position2': '兴隆堡', 'information': '未央-北窑站-兴隆堡 / 79.00㎡ / 南 / 2室1厅1卫 / 高阳湖  
1708 {'title': '整租·凤凰城 5居室 北卧', 'price': '770', 'position0': '碑林', 'position1': '黄雁村', 'position2': '凤凰城', 'information': '碑林-黄雁村-凤凰城 / 18.00㎡ / 北 / 5室0厅2卫 / 高阳湖  
1709 {'title': '整租·千禧社区 2室1厅 南/北', 'price': '2000', 'position0': '雁塔', 'position1': '曲江', 'position2': '千禧社区', 'information': '雁塔-曲江-千禧社区 / 90.00㎡ / 南 北 / 2室1厅1卫  
1710 {'title': '整租·民乐园万达广场 1室1厅 南/北', 'price': '7500', 'position0': '新城区', 'position1': '城内', 'position2': '民乐园万达广场', 'information': '新城区-城内-民乐园万达广场 / 120.00㎡ /  
1711 {'title': '整租·万科翡翠天麓 4室0厅 南/北', 'price': '23500', 'position0': '雁塔', 'position1': '鱼化寨', 'position2': '万科翡翠天麓', 'information': '雁塔-鱼化寨-万科翡翠天麓 / 385.50㎡ / 南  
1712 {'title': '整租·中海紫御华府 4居室 北卧', 'price': '690', 'position0': '雁塔', 'position1': '曲江', 'position2': '中海紫御华府', 'information': '精选 / 雁塔-曲江-中海紫御华府 / 15.00㎡ / 北 /  
1713 {'title': '整租·龙阳悦园二期 5居室 北卧', 'price': '850', 'position0': '雁塔', 'position1': '鱼化寨', 'position2': '龙阳悦园二期', 'information': '精选 / 雁塔-鱼化寨-龙阳悦园二期 / 18.00㎡ /  
1714 {'title': '整租·龙阳悦园二期 0厅 南', 'price': '11280', 'position0': '雁塔', 'position1': '高新四路', 'position2': '龙阳悦园二期', 'information': '雁塔-高新四路-龙阳悦园二期 / 188.00㎡ / 南 / 1室0厅0卫  
1715 {'title': '整租·龙阳悦园二期 5居室 南卧', 'price': '680', 'position0': '雁塔', 'position1': '鱼化寨', 'position2': '龙阳悦园二期', 'information': '精选 / 雁塔-鱼化寨-龙阳悦园二期 / 15.00㎡ /  
1716 {'title': '整租·招商雍丝路上 1室1厅 LOFT 北', 'price': '2300', 'position0': '莲桥', 'position1': '国际港务区', 'position2': '招商雍丝路上', 'information': '莲桥-国际港务区-招商雍丝路上 / 18.00㎡ /  
1717 {'title': '整租·大雁塔 1室1厅 南', 'price': '7000', 'position0': '雁塔', 'position1': '城西', 'position2': '大雁塔', 'information': '雁塔-城西-大雁塔 / 61.52㎡ / 南 / 1室1厅1卫 / 中庭  
1718 {'title': '整租·万科翡翠 3室1厅 南/北', 'price': '2700', 'position0': '莲桥', 'position1': '国际港务区', 'position2': '万科翡翠', 'information': '莲桥-国际港务区-万科翡翠 / 126.80㎡ / 南 北 /  
1719 {'title': '整租·紫薇田园都市0区 5居室 南卧', 'price': '1150', 'position0': '长安', 'position1': '紫薇田园都市', 'position2': '紫薇田园都市0区', 'information': '精选 / 长安-紫薇田园都市-紫薇田园
```

第二部分：数据存储

分别执行五个爬虫 beijing ,shanghai,guangzhou ,shenzhen ,xian 后，得到五个已经写入数据的 json 文件，即完成了储存部分。

	scrapy-beijing-zufang.json	2023/12/9 18:43	JSON File	766 KB
	scrapy-guangzhou-zufang.json	2023/12/9 18:51	JSON File	755 KB
	scrapy-shanghai-zufang.json	2023/12/9 18:47	JSON File	755 KB
	scrapy-shenzhen-zufang.json	2023/12/9 18:49	JSON File	758 KB
	scrapy-xian-zufang.json	2023/12/31 7:38	JSON File	771 KB

第三部分：数据处理与图表绘制

1. 比较 5 个城市的总体房租情况，包含租金的均价、最高价、最低价、中位数等信息，单位面积租金（元/平米）的均价、最高价、最低价、中位数等信息。采用合适的图或表形式进行展示。

1) 数据的处理：

从 json 文件中逐行读取数据。通过 json 文件可以得知，价格保存在列表的 price 项中，而面积保存在列表的 information 项中。通过 split 函数进行字符串分割，逐步分割得到需要的价格和面积。其中，面积取最小面积作为计算用的面积。

最后计算出各个城市的租金均价，最高价，最低价，中位数等信息，并将其保存到列表中。

with codecs.open(filename,'r',encoding='utf-8') as f:

```
# 打开并读取文件
read = f.readlines()
total_mid_price_count = []
space_mid_price_count = []
for index, info in enumerate(read): # 逐行读取
    data = json.loads(info)
    value = list(data.values())
    # value 保存了每一行的数据的值，以列表形式。
```

是小数

```
number = re.compile(r'^[-+]?[0-9]+\.[0-9]+$') # 正则式判断是否

# 划分出价格
temp_price = value[1].split('-')
price = temp_price[0]
price = int(price)
# 划分出面积
temp_space = value[5].split("m²")
temp_space = temp_space[0].split("/")
if len(temp_space) == 1:
    temp_space = temp_space[0].split("-")
    space = temp_space[0]
    space = float(space)
else:
    temp_space2 = temp_space[1].split("-")
    # 判断这个数是否为小数，是则说明是面积，这里保留了最小面积作为参考（否则就是异常数据，需要被忽略）
    result = number.match(temp_space2[0])
    if result:
        space = temp_space2[0]
        space = float(space)
    else: # 保留最小的面积（如 20-25 m²则将 space 看做 20）
        temp_space = temp_space[2].split("-")
        space = temp_space[0]
        space = float(space)
# print(space)
if price > temp_total_high_price: # 最高价
    temp_total_high_price = price
if price < temp_total_low_price: # 最低价
    temp_total_low_price = price
temp_total_price += price # 总价,均价出循环了计算
total_mid_price_count.append(price) # 中位数，同样出循环了计算

space_price = float(price/space)
if space_price > temp_space_high_price: # 最高价（单位面积）
    temp_space_high_price = space_price
if space_price < temp_space_low_price: # 最低价（单位面积）
    temp_space_low_price = space_price
temp_space_price += space_price # 总价（单位面积）
space_mid_price_count.append(space_price) # 中位数（单位面积）

# 均价
total_avg_price.append(float(temp_total_price/3000))
```



```

space_avg_price.append(float(temp_space_price/3000))
# 最高价
total_high_price.append(float(temp_total_high_price))
space_high_price.append(float(temp_space_high_price))
# 最低价
total_low_price.append(float(temp_total_low_price))
space_low_price.append(float(temp_space_low_price))
# 中位数
total_mid_price_count.sort()
space_mid_price_count.sort()
total_mid_price.append(float(total_mid_price_count[1499]))
space_mid_price.append(float(space_mid_price_count[1499]))

```

最终，得到如下所示的列表，列表共有五个元素，按顺序分别属于北京，广州，上海，深圳，西安：

```

C:\users\LEGIUN\AppData\Local\Microsoft\WindowsApps\python3.11.exe E:\Python\pacnong\zutang\total_price_snow
[8944.408333333333, 4069.3813333333333, 8065.405333333333, 6577.5243333333334, 4764.355]
[130000.0, 130000.0, 158000.0, 147000.0, 300000.0]
[1300.0, 200.0, 800.0, 899.0, 150.0]
[6600.0, 2600.0, 5500.0, 4500.0, 2100.0]
[103.21263917979654, 63.876643122690574, 103.47060846736892, 88.50262501132634, 40.48050069027929]
[1098.265895953757, 516.0, 430.7692307692308, 529.5389048991354, 450.0]
[15.729865771812081, 6.282283959234958, 12.95489341655871, 22.321428571428573, 5.925925925925926]
[98.66327180140037, 50.22601707684581, 96.42778873548104, 78.84615384615384, 32.5]
进程已结束，退出代码为 0

```

2) 图表的绘制。

采用 matplotlib.pyplot 来进行绘制：

对于该数据，采用直方图的形式进行展示，共八个子图，分为两行四列。使用 plt.bar 进行直方图的绘制。

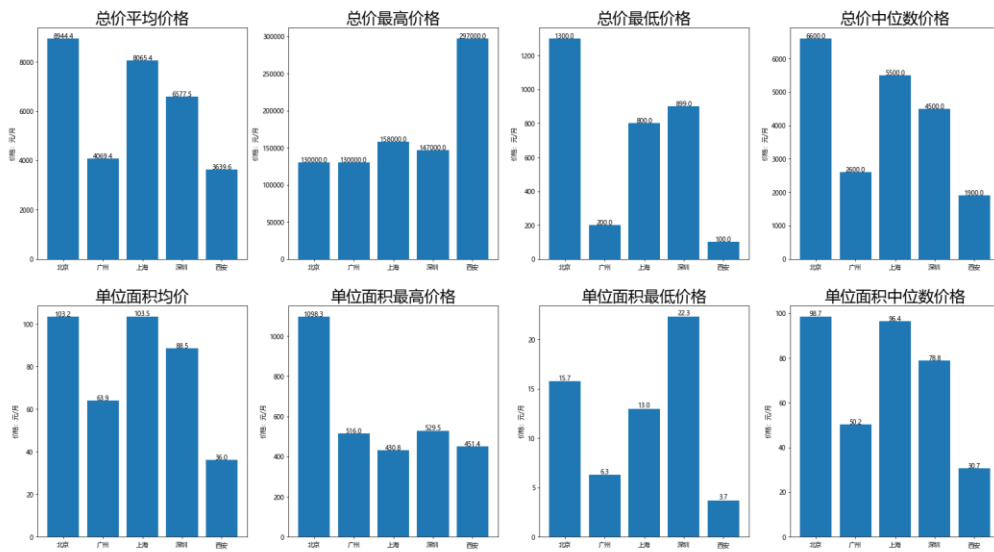
其中，一个子图的绘制代码如下：

```

# 绘直方图图
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
city_name = ['北京', '广州', '上海', '深圳', '西安']
plt.figure(figsize=(30, 30), dpi=70)
bar_width = 0.4

# 总平均租金展示图的绘制，共八个子图
plt.subplot(241)
plt.title('总价平均价格', fontsize=25)
plt.bar(city_name, total_avg_price)
plt.ylabel('价格：元/月')
for x, y in enumerate(total_avg_price):
    plt.text(x, y + 0.1, "%s" % round(y, 1), ha='center')

```



2. 比较 5 个城市一居、二居、三居的情况，包含均价、最高价、最低价、中位数等信息。

1) 数据的处理：

在 json 文件中可以看到居室的信息保存在 information 类中，其中为形如“1 室 1 厅 1 卫”中的“1 室”即为居室情况。通过 split 函数进行划分，得到需要的部分，然后按居室的类别进行价格的计算和存储。同时，由于要比较五个城市的三种类型的居室，因此采用了嵌套列表的数据结构来存储相应的居室。存储的结构为[[北京 1 居, 广州 1 居, 上海 1 居, 深圳 1 居, 西安 1 居],[北京 2 居, 广州 2 居, 上海 2 居, 深圳 2 居, 西安 2 居],[北京 3 居, 广州 3 居, 上海 3 居, 深圳 3 居, 西安 3 居]]

列表初始化如下：

```
avg_price = [[]for _ in range(3)] # 均价
high_price = [[]for _ in range(3)]# 最高价
mid_price = [[]for _ in range(3)]# 中位数
low_price = [[]for _ in range(3)]# 最低价
```

数据处理部分如下：

```
with codecs.open(filename, 'r', encoding='utf-8') as f:
    read = f.readlines()
    for index, info in enumerate(read):
        data = json.loads(info)
        value = list(data.values())
        # value 保存了每一行的数据的值，以列表形式。
        # 数据处理，拆分出价格
        temp_price_read = value[1].split('-')
        price = temp_price_read[0]
        price = int(price)
        # 数据处理，拆分出居室情况
        temp_house_type = value[5].split('室')
```

```

temp_house_type = temp_house_type[0].split('/')
if temp_house_type[len(temp_house_type)-1] == '1':
    house_type = 1
if temp_house_type[len(temp_house_type)-1] == '2':
    house_type = 2
if temp_house_type[len(temp_house_type)-1] == '3':
    house_type = 3
# print(house_type)

if house_type == 1: # 1 室的情况
    count[0] += 1
    if price > temp_high_price[0]: # 最高价
        temp_high_price[0] = price
    if price < temp_low_price[0]: # 最低价
        temp_low_price[0] = price
    temp_price[0] += price
    temp_mid_price_count[0].append(price)

if house_type == 2: # 2 室的情况
    count[1] += 1
    if price > temp_high_price[1]: # 最高价
        temp_high_price[1] = price
    if price < temp_low_price[1]: # 最低价
        temp_low_price[1] = price

    temp_price[1] += price
    temp_mid_price_count[1].append(price)

if house_type == 3: # 3 室的情况
    count[2] += 1
    if price > temp_high_price[2]: # 最高价
        temp_high_price[2] = price
    if price < temp_low_price[2]: # 最低价
        temp_low_price[2] = price
    temp_price[2] += price
    temp_mid_price_count[2].append(price)

for i in range(0, 3):
    # 将处理完毕的数据放入列表中
    temp_price[i] = float(temp_price[i]/count[i]) # 均价
    avg_price[i][city] = temp_price[i]
    high_price[i][city] = temp_high_price[i]
    low_price[i][city] = temp_low_price[i]
    temp_mid_price_count[i].sort()

```

```
mid_price[i][city]
=temp_mid_price_count[i][int(len(temp_mid_price_count[i])/2)]
```

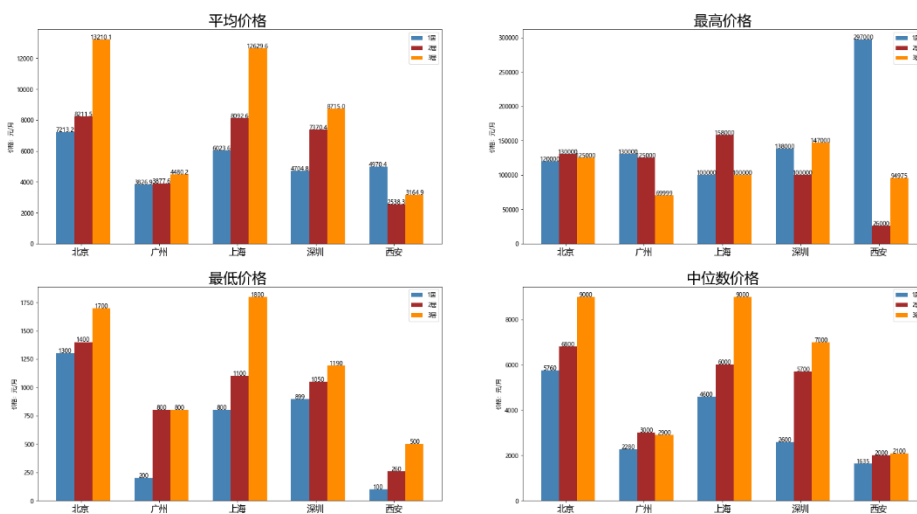
```
C:\Users\LEGION\AppData\Local\Microsoft\WindowsApps\python3.11.exe E:\Python\pachong\zuofang\room_price_show.py
[[{"city": "北京", "type": "1居", "price": 7213}, {"city": "北京", "type": "2居", "price": 8914}, {"city": "北京", "type": "3居", "price": 13210}, {"city": "广州", "type": "1居", "price": 3614}, {"city": "广州", "type": "2居", "price": 3777}, {"city": "广州", "type": "3居", "price": 4302}, {"city": "上海", "type": "1居", "price": 6221}, {"city": "上海", "type": "2居", "price": 8024}, {"city": "上海", "type": "3居", "price": 14248}, {"city": "深圳", "type": "1居", "price": 5243}, {"city": "深圳", "type": "2居", "price": 6124}, {"city": "深圳", "type": "3居", "price": 7113}, {"city": "西安", "type": "1居", "price": 4024}, {"city": "西安", "type": "2居", "price": 4318}, {"city": "西安", "type": "3居", "price": 5143}]]
```

2) 图表的绘制:

采用直方图的形式进行展示,但是对于每个城市,有三种居室的情况要展示,因此在绘制直方图的基础上,要让一个 X 坐标数据显示出三个类型的图像。因此,设置一个图像宽度,通过 bar 函数的参数来设置绘制多个直方图。

```
plt.subplot(221)
plt.title('平均价格 ', fontsize=25)
# 绘制成一个 X 坐标对应多个直方图的, 这样方便数据的对比和展示观看
plt.bar(x=np.arange(len(city_name)), height=avg_price[0], width=bar_width, label='1 居', color='steelblue')
plt.bar(x=np.arange(len(city_name))+bar_width, height=avg_price[1], width=bar_width, label='2 居', color='brown')
plt.bar(x=np.arange(len(city_name))+bar_width*2, height=avg_price[2], width=bar_width, label='3 居', color='darkorange')
plt.xticks(np.arange(5)+0.2, city_name, fontsize=15)
plt.ylabel('价格: 元/月')
# 绘制图例
plt.legend()
# 给图像上端增添数据显示
for x, y in enumerate(avg_price[0]):
    plt.text(x, y + 0.2, "%s" % round(y, 1), ha='center')
for x, y in enumerate(avg_price[1]):
    plt.text(x+bar_width, y + 0.2, "%s" % round(y, 1), ha='center')
for x, y in enumerate(avg_price[2]):
    plt.text(x+bar_width*2, y + 0.2, "%s" % round(y, 1), ha='center')
```

绘图结果如下:



3. 计算和分析每个城市不同板块的均价情况，并采用合适的图或表形式进行展示。



如图所示的“城西”即为板块名称。

1) 数据的处理：

板块名称位于 information 中，同时，由于并不是每一条数据都包含板块，因此要对不包含板块的数据进行剔除，对包含板块的数据则与预先建立好的板块列表进行比较，若板块名已经存在在板块列表当中，则在该板块名的统计数上+1。当遍历完所有结果后，再根据板块名称和种类数量进行各项价格的计算。

python 代码：

```
with codecs.open(filename, 'r', encoding='utf-8') as f:
    read = f.readlines()
for index, info in enumerate(read):
    data = json.loads(info)
    value = list(data.values())
    # value 保存了每一行的数据的值，以列表形式。
    # 划分出价格
    temp_price_read = value[1].split('-')
    price = temp_price_read[0]
    price = int(price)
    # 划分出板块
    temp_pos_read = value[5].split('-')
    if len(temp_pos_read) > 1: # 先确定板块的位置，再将干扰的数据剔除
        temp_pos_read = temp_pos_read[1].split('m²')
        if len(temp_pos_read) == 1:
            pos = str(temp_pos_read[0])
            # 通过遍历的方式来检查已经建立的板块名列表，重复则数量加一，
            # 无重复则将这个板块名加入板块列表
            flag = -1
            for j in range(len(pos_name)):
                if pos_name[j] == pos or pos_name[j] == "":
                    flag = j
                    break

            pos_name[flag] = pos
            count[flag] += 1
            temp_price[flag] += price
    # 将之前预设的列表里多余的空项去除掉
    while temp_price[-1] == 0:
        temp_price.pop()
```

得到如下的结果:

2) 图表的绘制:

```
for i in range(5):
    while len(avg_price[i]) > 15:
        avg_price[i].pop()
    while len(pos_list[i]) > 15:
        pos_list[i].pop()
```

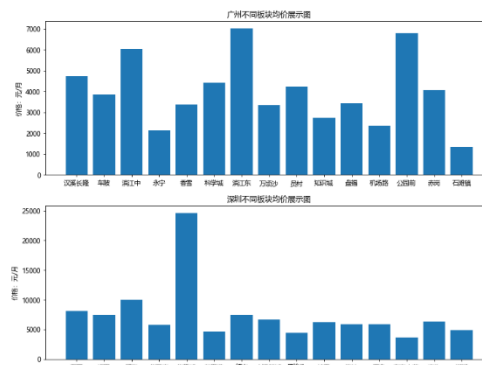
```
# 北京
plt.subplot(321)
plt.bar(pos_list[0],avg_price[0])
```



```
plt.ylabel("价格：元/月")
```

The figure consists of three bar charts, each representing a different city: Beijing (top), Shanghai (middle), and Wuhan (bottom). Each chart displays the average monthly price for 14 different types of non-woven fabrics. The y-axis for all charts is labeled '价格: 元/月' (Price: Yuan/month) and ranges from 0 to 2500 for Beijing, 0 to 1750 for Shanghai, and 0 to 7000 for Wuhan. The x-axis for each chart lists the fabric types: 大编织 (Large woven), 涤纶絮 (Polyester絮), 长丝 (Long filament), 芳纶 (Aramid), 聚丙烯 (Polypropylene), 涤纶 (Polyester), 无纺布 (Non-woven fabric), 无纺布 (Non-woven fabric), 无纺布 (Non-woven fabric), 无纺布 (Non-woven fabric), 无纺布 (Non-woven fabric), 无纺布 (Non-woven fabric), 无纺布 (Non-woven fabric), 无纺布 (Non-woven fabric).

城市	大编织	涤纶絮	长丝	芳纶	聚丙烯	涤纶	无纺布	无纺布	无纺布	无纺布	无纺布	无纺布	无纺布	无纺布
北京	1000	700	600	700	800	900	700	900	700	2500	1300	900	700	500
上海	400	1200	900	1200	800	800	600	700	700	1700	800	1600	600	600
武汉	3000	7000	3000	2000	3000	3000	3000	3000	2000	4000	6000	2000	3000	2000



1) 数据的处理:

```
with codecs.open(filename, 'r', encoding='utf-8') as f:
```

```
read = f.readlines()
```

```
# 打开文件并逐行读取
```

```
for index, info in enumerate(read):
```

```
data = json.loads(info)
```

```
value = list(data.values())
```

value 保存了每一行的数据的值，以列表形式。

不断拆分，最后拆出来需要的面向

```
temp_price_read = value[1].split('-')
```

```
price = temp_price_read[0]
```

```
price = int(price)
```

```
temp_face_read = value[5].split('m2 /')
```

```
temp face read = temp face read[1].split(' / ')
```

```
temp_face_read = temp_face_read[0].split('在')
```

```
if len(temp_face_read) == 1:
```

```
temp_face_read = temp_face_read[0].split('室')
```

```
if len(temp_face_read) == 1:
    temp_face_read = temp_face_read[0].split(' ')
    # 由于同时一组数据可能有多个面向，故如面向东 南，则东，南均
    各统计一次。
```

```
for face in temp_face_read:
    if face == '东':
        temp_price[0] += price
        count[0] += 1
    if face == '南':
        temp_price[1] += price
        count[1] += 1
    if face == '西':
        temp_price[2] += price
        count[2] += 1
    if face == '北':
        temp_price[3] += price
        count[3] += 1

# 计算平均价格
for i in range(len(count)):
    temp_price[i] = temp_price[i]/count[i]

# 将价格放入列表中
avg_price.append(temp_price)
count_list.append(count)
```

```
C:\Users\LEI\OneDrive\Local\Microsoft\WindowsApps\python3.11.exe E:\Python\pachong\zufang\face_price_show.py
[[8432, 432642487067, 9551, 949874934316, 9416, 701257861616, 10455, 941415785192], [6782, 8086745762715, 4881, 169483610078, 9387, 752475247526, 5372, 558878584673], [9159, 402985074626, 8879, 763205011429, 9906, 730158730159, 9163,
[[186, 1983, 318, 1229], [236, 1157, 181, 535], [87, 2234, 63, 427], [146, 958, 71, 289], [229, 2115, 165, 1078]]]
```

2) 图表的绘制:

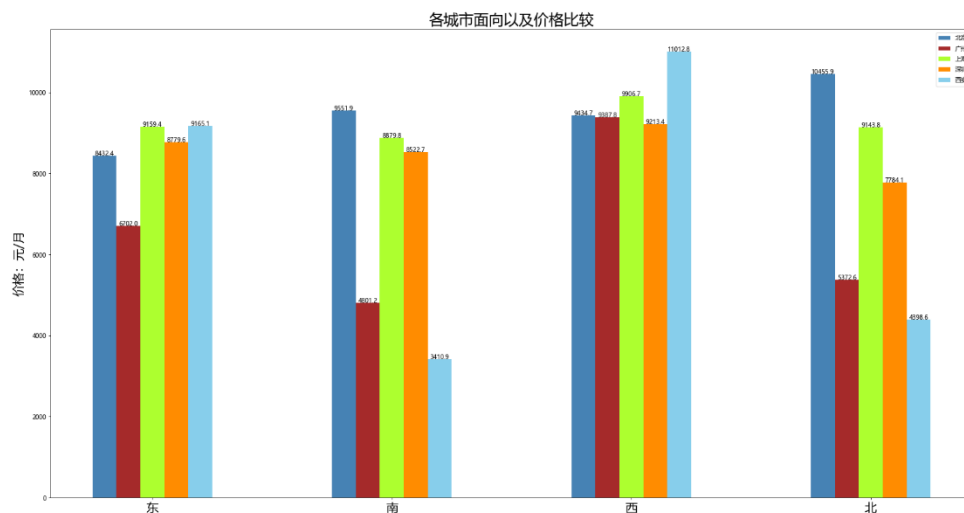
通过 X 坐标偏移来绘制叠加的直方图，使用 plt.bar 函数，并同时展示图例和在上方显示具体数字。设置 X 坐标为四种面向东，南，西，北。每个面向分别显示五个城市的均价。

绘图，分别将五个城市的直方图进行绘制

```
face_name = ['东','南','西','北']
plt.figure(figsize=(30, 30), dpi=70)
bar_width = 0.1 # 条宽偏移
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']

# 通过偏移来绘制直方图，以达到一个横坐标能显示多个直方图的效果，同时便于区分颜色和图例
plt.title('各城市面向以及价格比较', fontsize=25)
plt.bar(x=np.arange(len(face_name)), height=avg_price[0], width=bar_width, label='北京', color='steelblue')
plt.bar(x=np.arange(len(face_name)) + bar_width, height=avg_price[1], width=bar_width, label='广州', color='brown')
plt.bar(x=np.arange(len(face_name)) + bar_width * 2, height=avg_price[2], width=bar_width, label='上海', color='greenyellow')
```

```
plt.bar(x=np.arange(len(face_name)) + bar_width * 3, height=avg_price[3],
width=bar_width, label='深圳',color='darkorange')
plt.bar(x=np.arange(len(face_name)) + bar_width * 4, height=avg_price[4],
width=bar_width, label='西安',color='skyblue')
plt.xticks(np.arange(4) + 0.2, face_name,fontsize=20)
plt.ylabel('价格：元/月',fontsize=20)
绘制生成的图形如图所示：
```



5. 查询各个城市的人均 GDP，分析并展示其和单位面积租金分布的关系。

1) 数据处理：

#通过百度查询各个城市的人均 GDP 可知：

人均 GDP：北京：19.03 万元/人 广州：15.36 万元/人 上海：17.99 万元/人 深圳：18.33 万元/人 西安：8.88 万元/人

因此，直接使用 list 赋初值的形式存储各个城市的人均 GDP。同时，由于在前面的程序中已经计算出单位面积租金，因此直接引用并给数组赋初值。为了在后面的可视化中能够更加形象的展示出 GDP 和单位面积租金的关系并方便分析，我计算了单位面积租金和人均 GDP 的比值，并存在 list 中，这样就可以使数据更加明了。

```
GDP = [19.03, 15.36, 17.99, 18.33, 8.88]
```

```
space_price = [103.2, 63.9, 103.5, 88.5, 36.0]
```

```
count = []
```

求比值

```
for i in range(0,5):
```

```
    count.append(space_price[i]/GDP[i])
```

2) 图表绘制

绘制两个图表，分别为比值的直方图和单位面积租金-人均 GDP 的散点图。通过使用 plt.bar 函数和 plt.scatter 来进行绘制，在绘制散点图的过程中，由于要绘制五个城市并在点的颜色上进行区分，因此五次单独调用绘制散点图的函数，这样方便区分不同的城市。

绘制直方图

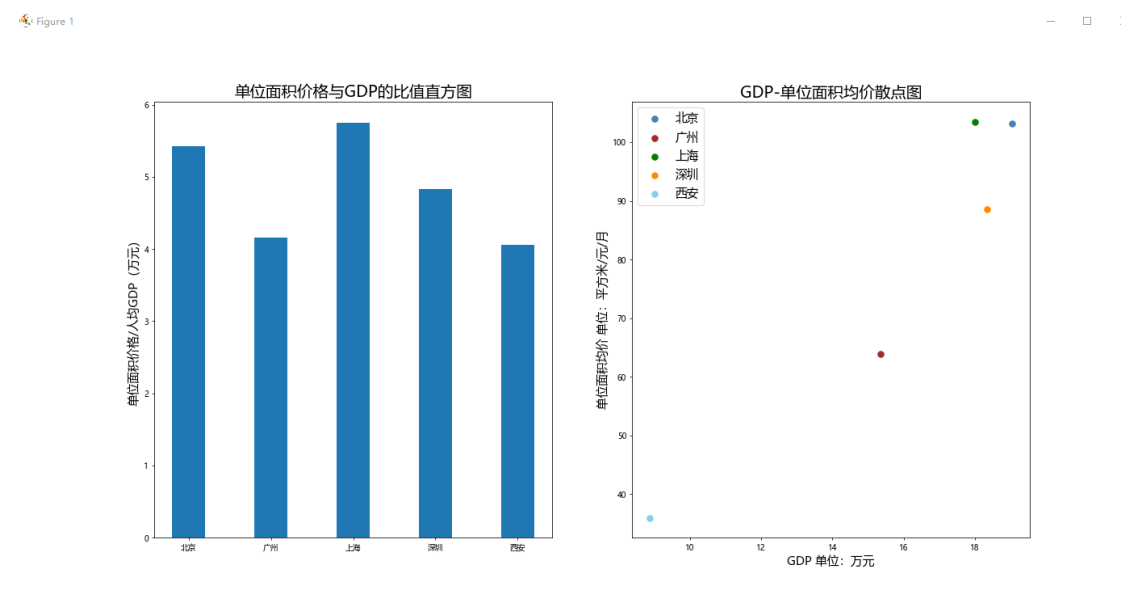
```
plt.subplot(121)
```

```
plt.title("单位面积价格与 GDP 的比值直方图",fontsize=20)
```

```
plt.bar(city_name, count,width=0.4)
plt.ylabel("单位面积价格/人均 GDP（万元）",fontsize=15)

# 绘制散点图
plt.subplot(122)
plt.scatter(GDP[0],space_price[0],s=60, label='北京', color='steelblue')
plt.scatter(GDP[1],space_price[1],s=60, label='广州', color='brown')
plt.scatter(GDP[2],space_price[2],s=60, label='上海',color='green')
plt.scatter(GDP[3],space_price[3],s=60, label='深圳',color='darkorange')
plt.scatter(GDP[4],space_price[4],s=60, label='西安',color='skyblue')
plt.title("GDP-单位面积均价散点图",fontsize=20)
plt.xlabel("GDP 单位： 万元",fontsize=15)
plt.ylabel("单位面积均价 单位： 平方米/元/月",fontsize=15)
plt.legend(fontsize=15)
```

绘图结果为：



6. 查询各个城市的平均工资，分析并展示其和单位面积租金分布的关系。

1) 数据处理：

#查询百度各个城市的人均工资可知：

#人均工资： 北京：13567 元/月 广州：11300 元/月 上海：12183 元/月 深圳：12300 元/月 西安 9011 元/月。对查询的结果直接使用 list 赋初值的形式进行存储（由于人均工资数目较大，因此将单位改为百元，即每个数据都除以 100），同时由于上面已经计算了单位面积均价，也通过数组直接赋初值的形式进行存储。为了方便后续的数据分析，我还计算了单位面积均价和人均工资的比值，比值越小，说明租金占工资比重越小。

```
space_price = [103.2, 63.9, 103.5, 88.5, 36.0]
```

```
salary = [135.67, 113.00, 121.83, 123.00, 90.11] # 单位为百元，这样好计算一点
```

```
count = []
```

```
# 计算比值
```

```
for i in range(0,5):
```

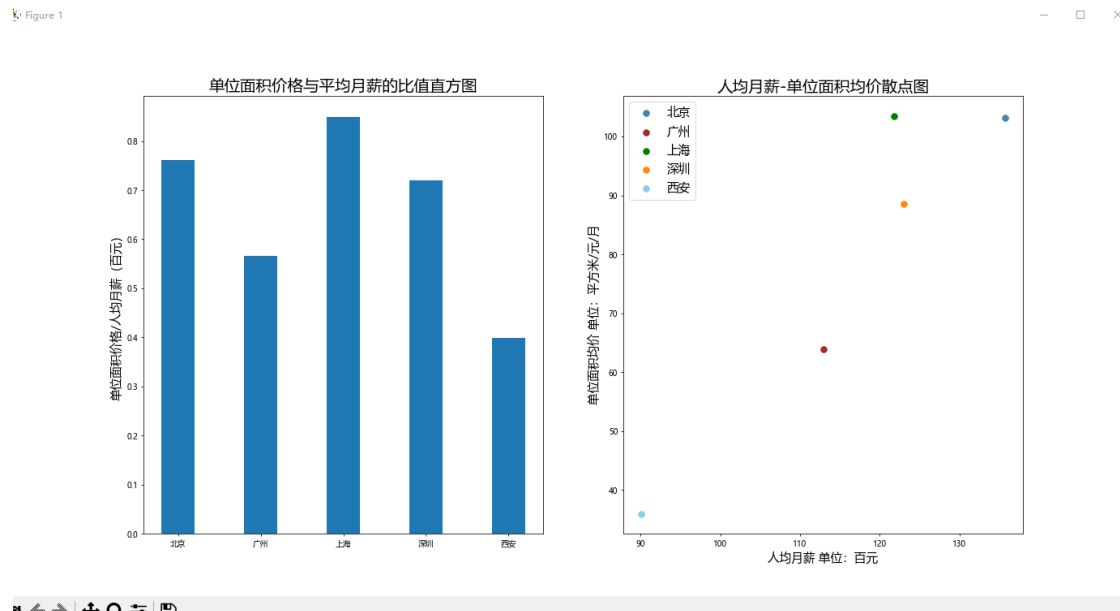
```
    count.append(space_price[i]/salary[i]) # 用比值来表示，比值越低说明房租占工资占比
```

小，生活成本相对低一点

2) 图表绘制：

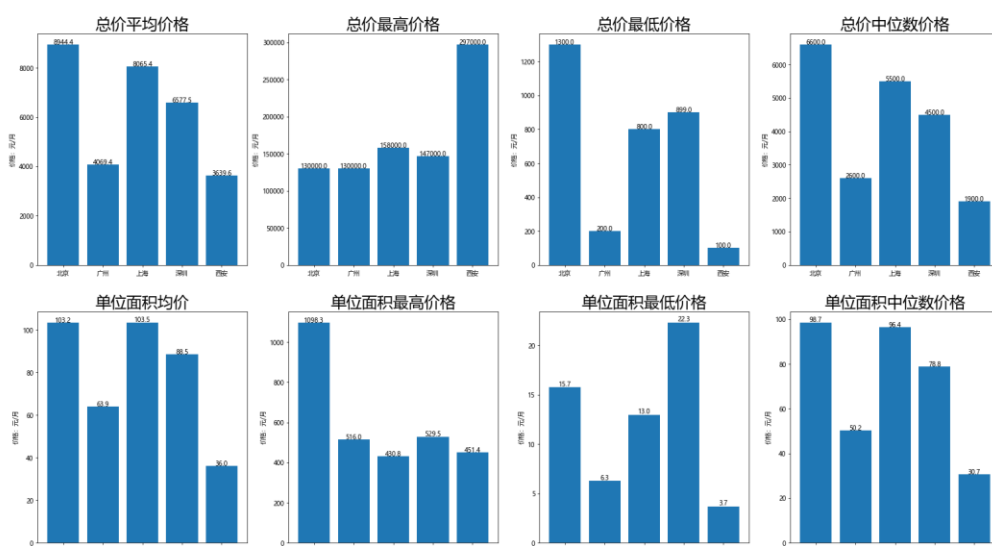
绘制两个图表，分别为比值的直方图和单位面积租金-人均月工资的散点图。通过使用 plt.bar 函数和 plt.scatter 来进行绘制，在绘制散点图的过程中，由于要绘制五个城市并在点的颜色上进行区分，因此五次单独调用绘制散点图的函数，这样方便区分不同的城市。

绘图结果：



三、数据分析

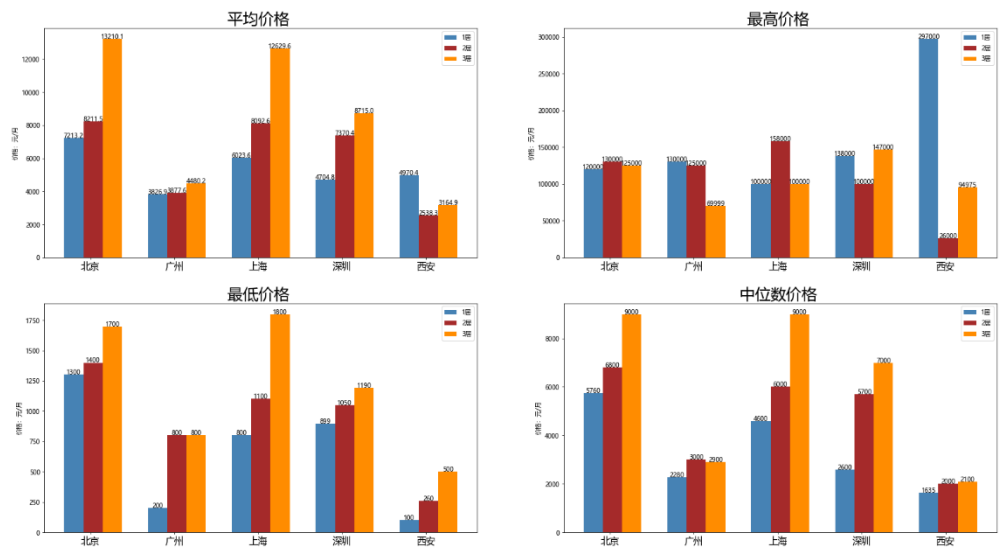
1.5 个城市的整体租房情况比较和分析：



从图表中可以看到，总平均价格和单位面积均价对于五个城市来说趋势是一样的：最高的为北京，其次为上海，接下来是深圳，之后是广州，最后是西安。这也符合我们的生活常识，北上广深均为一线城市，尤其北京作为首都，房屋租金价格都普遍偏高，而广州在一线城市中不如上海和深圳那么经济发达，因此租金也相对较低一些。而与之相比，西安作为一个二

线城市，而且为中国中西部地区的城市，房屋租金均价甚至不如北京的一半。
但是，在总最高价格中，西安也出现了一项例外的很高的数据，因此导致了这一部分图表的统计中西安变成了最高的。对于中位数的趋势，也和均价的趋势是差不多的，因此可以分析得出结论，整体租房价格北京>上海>深圳>广州>西安。

2.5 个城市一居、二居、三居的租金价格情况分析：



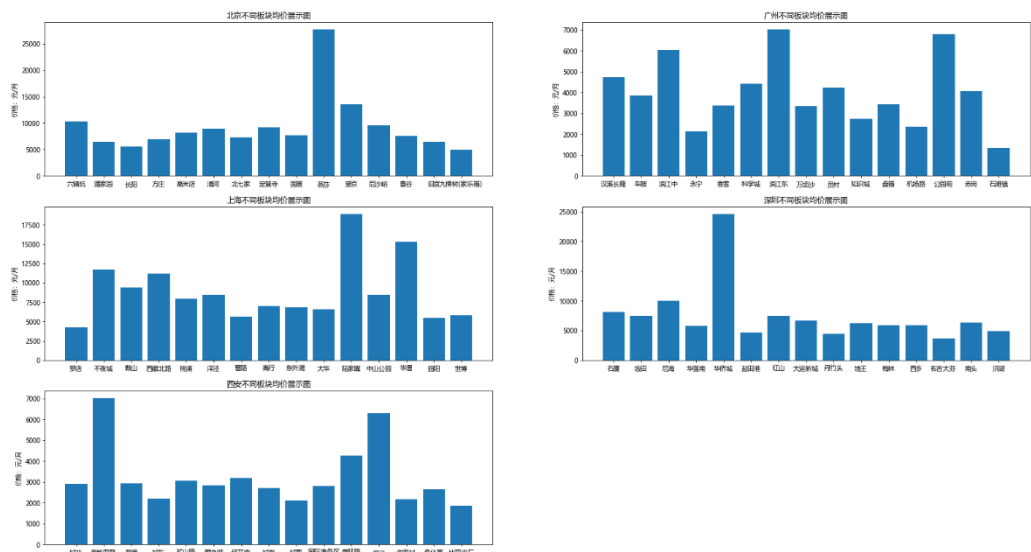
图表中每个城市都有三条直方图，从左向右依次为一居室，二居室，三居室。
从图中可以看出，对于均价和中位数价格这两个能够较为客观反应租房整体情况的数据。在居室数量上五个城市的趋势大体上均为居室越多，租金越贵。从最低价格上也能反应出这一点。当然，由于最高价和最低价都可能存在一些特例，因此在一定程度上会导致统计的偏差。这也符合我们的客观认知，居室越多，说明房屋越大，相应的房屋质量可能也就偏好一些，因此租金价格会稍微贵一些。同时，在各个城市之间，租金的价格趋势也和上面分析的一样，北京>上海>深圳>广州>西安。但对于西安来说，在均价上出现了一居>三居>二居的情况。我猜测这个可能原因是因为西安相对于北上广深处于西部地区，土地价格也相较于便宜一些，因此开发商建楼盘的时候并没有四个一线城市那么寸土寸金，多以多居室为主，很少建单间的房屋，因此导致1居室的数量比2,3居室的数量相较于北上广深少得多。

3. 分析每个城市不同板块的均价情况。

在分析数据的时候，我顺便统计了五个城市各自的板块数量，如下图所示。

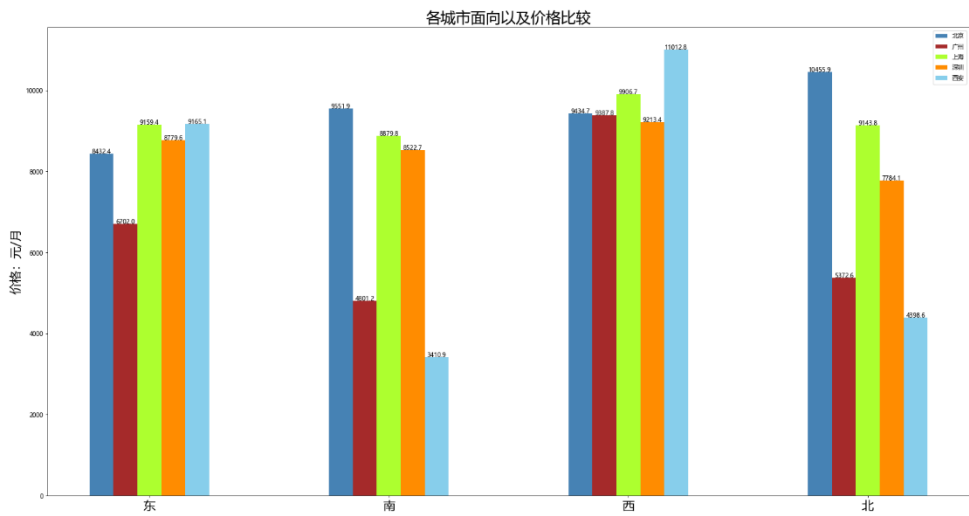
[234, 217, 177, 89, 49]

从左至右依次为北京，广州，上海，深圳，西安。可以看到对于每个城市其实板块数目都很多，尤其是北京有两百多个不同的板块，板块名并不方便在一张图表中进行展出，因此我只截取了每个城市前十五个板块的均价数据进行展示，这样也能从局部观察和分析各个城市板块的情况。



首先对于每个城市，都能看到有那么一两个均价相较于其他板块很高的板块，比如北京的燕莎，广州的滨江东，上海的陆家嘴，深圳的华侨城，西安的高新四路和曲江。其中，陆家嘴是大家众所周知的上海金融中心，因此该板块的租金肯定是比较贵的。对于深圳和广州这两个城市我不了解，但对于西安的高新四路和曲江两个板块来说，高新四路为西安最有名最好的高中之一的一个高中-高新一中的地址，同时也是学费很贵的一所学校，在该校上学的学生多为周围县市区的尖子生，因此有着不小的租房需求，而且该地区同时为西安的开发区，经济相对来说也比较发达，学校周围多为商业写字楼，写字楼中有很多改建过的在出租的房子，价格相较于其他地区来说偏贵是理所当然的。对于曲江板块，则由于该地段为西安几个较贵的楼盘的聚集地，是西安居民心中各种意义上的“富人区”，附近有曲江公园等较好的城市环境，和位于南二环的便利交通，因此该板块的房屋租金也偏高也是合情合理的。

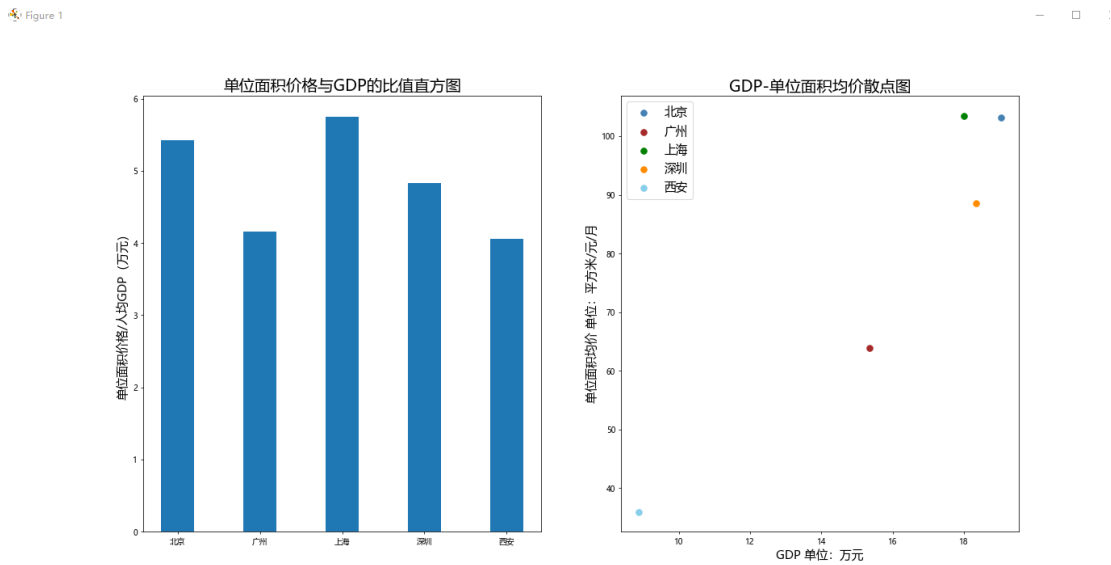
4. 比较各个城市不同朝向的单位面积租金分布情况，哪个方向最高，哪个方向最低？各个城市是否一致？如果不一致，你认为原因是什么？



在这张图表中，X 轴分别为东，南，西，北，价格从左至右五个柱状图分别为北京，广州，

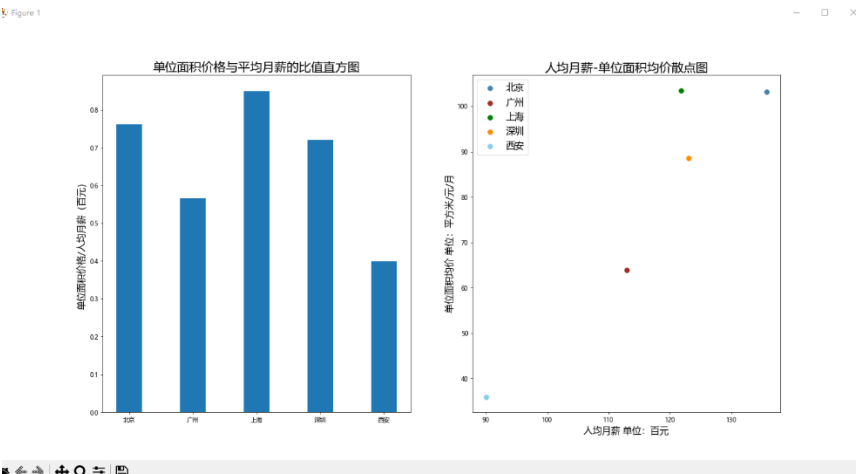
上海，深圳，西安。从图中可以看出，面向西边的租金整体价格是最高的，面向南边的租金是整体价格最低的，我认为这个是由于中国房屋建筑大多讲究“坐北朝南”，因此面向南边的房间会较多，房屋基数上去了，租金可能也就相应的低一些。除了北京其他四个城市均符合这一规律。而北京出现了偏差，为面东的最低，面北的最高。我猜测可能的原因是北京在五个城市中地理位置最偏北，因此面北的房屋太阳直晒的时间最少，同时北边有燕山等，在天气晴朗的时候能看见，景观相对来说也比较好，而面东的房屋可能湿气较重一些，因为北京东边是海的缘故。当然，也有可能只是偶然性的偏差，总体趋势还是面南的比较便宜。

5. 分析并展示各个城市 GDP 和单位面积租金分布的关系。相对而言，在哪个城市租房的性价比最高？



第一张图展示了单位面积均价和 GDP 的比值。当单位面积一定时，GDP 越大，说明该单位面积能造成的生产价值越多，因此租房也就更有性价比，因此，单位面积均价/GDP 的比值越小越好。从图 1 中可以看出，西安的比值是最低的，其次是广州和西安差不多，之后是深圳，北京，最后是上海。在散点图中是同理的，点与原点连线的斜率越小说明性价比越高。因此可以得出结论租房性价比西安>广州>深圳>北京>上海。

6. 分析并展示各个城市人均工资和单位面积租金分布的关系。相对而言，在哪个城市租房的负担最重？



对于图 1，当工资相同时，单位面积均价越大，说明租房开销在工资的占比越大，租房的负担越重，因此，单位面积均价/人均工资的比值越大说明租房负担越大，从图中可以看出，租房负担上海>北京>深圳>广州>西安。对于散点图来说，点与原点连线的斜率越大，说明租房负担越重，因此也可以得出结论，租房负担：上海>北京>深圳>广州>西安。

四、实验结论

通过本次实验，我对北京，上海，广州，深圳，西安这五个城市的租房数据进行了爬取，处理，可视化展示，对比，分析。最终得到结论，租房租金不论是从价格还是性价比亦或是负担程度上来说，二线城市的负担都比一线城市要小一些，也更适合一些。这也是为什么人们说一线城市的生活压力更大。

在本次实验中，首先我通过 scrapy 爬虫框架爬取了链家官网的数据，在之前构建的爬取链家新房数据的爬虫小作业的基础上对代码进行了改动，并成功执行了爬虫爬取到了五个城市的数据，存储在了 json 文件中，整个爬取的过程由于有了之前的经验，因此比较顺利，只在修改 xpath 中遇到了一定的困难，因为每个租房信息的 information 的部分构成是不一样的，有些会缺少一些数据，因此造成了一定的定位困难，但后续我通过整体存储的方式解决了这一问题。

之后对于数据的读取和分析部分，由于也有了之前数据可视化作业的经验，对于如何通过 split 函数来划分出需要的字段和如何使用 plt 绘图函数来进行各种图像的绘制，都有一定的基础，因此实现过程也较为顺利，除了在处理居室的时候处理嵌套列表部分出现了一些数据修改上的问题，后来通过查询相关资料解决，之后我又去查阅了 plt 绘图函数的各种参数，使得绘制出来的图表更加明晰和展示的清晰，同时还绘制了一些多项直方图，并能通过颜色进行区分等。最后通过绘制出的图表，以及结合生活经验常识，对数据进行了分析并得到了相应的结论。

总的来说，通过本次实验，我提高了个人的编程能力，代码构建能力，问题查找和解决 BUG 能力，以及资料查询和学习新知识的能力，进一步掌握了爬虫工具，plt 绘图工具等有用的知识，并结合实际操作完成了相应的任务，也提高了对数据进行观察和分析的能力，对北上广深和西安这五个城市的租房相关信息有了一定的了解。极大的提高了个人综合能力。