先构造出获得经纬度函数便于调用

In [1]:

```
import requests,json
# 返回经纬度
def gain_location(adress):
    api_url=f'https://restapi.amap.com/v3/geocode/geo?city=北京市&address={adress}&key=自己的key&out
    r = requests.get(api_url)
    r = r.text
    r = r.strip('showLocation(')#高德
    r = r.strip(')')
    jsonData = json.loads(r)['geocodes'][0]['location'] # 将json字符串转换为字典类型转为字典格式类型
    return jsonData
```

构造路径规划函数

https://lbs.amap.com/api/webservice/guide/api/direction/
(https://lbs.amap.com/api/webservice/guide/api/direction/)

In [5]:

```
from PIL import Image
# import String

image1 = Image.open("1.png")
image1.show()
```

- **请求参数**

| 参数名 | 含义 | 规则说明 | 是否必须 | 缺省值 |
|---|---|---|---|---|
| key | 请求服务权限标识 | 用户在高德地图官网申请Web服务API类型KEY | 必填 | 无 |
| origin | 出发点 | 规则： lon，lat（经度，纬度），“,”分割，如117.500244,40.417801 经纬度小数点不超过6位 | 必填 | 无 |
| destination | 目的地 | 规则： lon，lat（经度，纬度），“,”分割，如117.500244,40.417801 经纬度小数点不超过6位 | 必填 | 无 |
| sig | 数字签名 | 数字签名获取和使用方法 | 可选 | 无 |
| output | 返回数据格式类型 | 可选值：JSON，XML | 可选 | JSON |
| callback | 回调函数 | callback值是用户定义的函数名称，此参数只在output=JSON时有效 | 可选 | 无 |

## 步行路径规划

- 接口简介：

步行路径规划 API 可以规划100KM以内的步行通勤方案，并且返回通勤方案的数据。最大支持 100km 的步行路线规划。

- 步行路径规划API URL

| | |
|---|---|
| URL | https://restapi.amap.com/v3/direction/walking?parameters |
| 请求方式 | GET |

parameters 代表的参数包括必填参数和可选参数。所有参数均使用和号字符(&)进行分隔。下面的列表枚举了这些参数及其使用规则。

### 在高德地图上面申请一个key -> 协作开发者


### 通过上面可以看出key是65a3a30da6a815fdf2dc1fc98bd48ba1

### 经纬度转换

In [27]:

```
import requests
import json

def get_location(county):  # 设置函数转换经纬度
    url = 'https://restapi.amap.com/v3/geocode/geo'  # 高德API地理编码服务地址
    params = {'key': '65a3a30da6a815fdf2dc1fc98bd48ba1',  # 参数1：个人申请的高德密钥
              'address': county } #参数2：需要转换经纬度的位置名称
    try:
        response = requests.get(url, params) #使用requests模块的get方法请求网址数据
        jd = json.loads(response.text)  #数据json格式化
        return jd['geocodes'][0]['location'] #读取需要的location值
    except:
        return  '未获取经纬度' #利用try-except设置防呆机制
```

In [28]:

```
get_location("北京")
```

Out[28]:

'116.407526,39.904030'

### 两地之间驾车距离的计算

In [11]:

```
import requests
import json

def get_distance(origin,destination): #设置函数计算两经纬度间驾车距离
    url = 'https://restapi.amap.com/v3/direction/driving'  # 高德API驾车路径规划服务地址
    params = {'key': '65a3a30da6a815fdf2dc1fc98bd48ba1',  # 参数1：个人申请的高德密钥
              'origin': origin, # 参数2：起始点的经纬度坐标
              'destination':destination, # 参数3：目的地的经纬度坐标
              'extensions':'base'} # 参数4：返回结果控制选项，必填项，base:返回基本信息；all：返回全
    try:
        response = requests.get(url, params)  #使用requests模块的get方法请求网址数据
        jd = json.loads(response.text)  #数据json化
        return jd['route']['paths'][0]['distance']  #读取需要的distance值
    except:
        return  0  #利用try-except设置防呆机制，这里设置距离0表示未成功获取两地间的距离
```

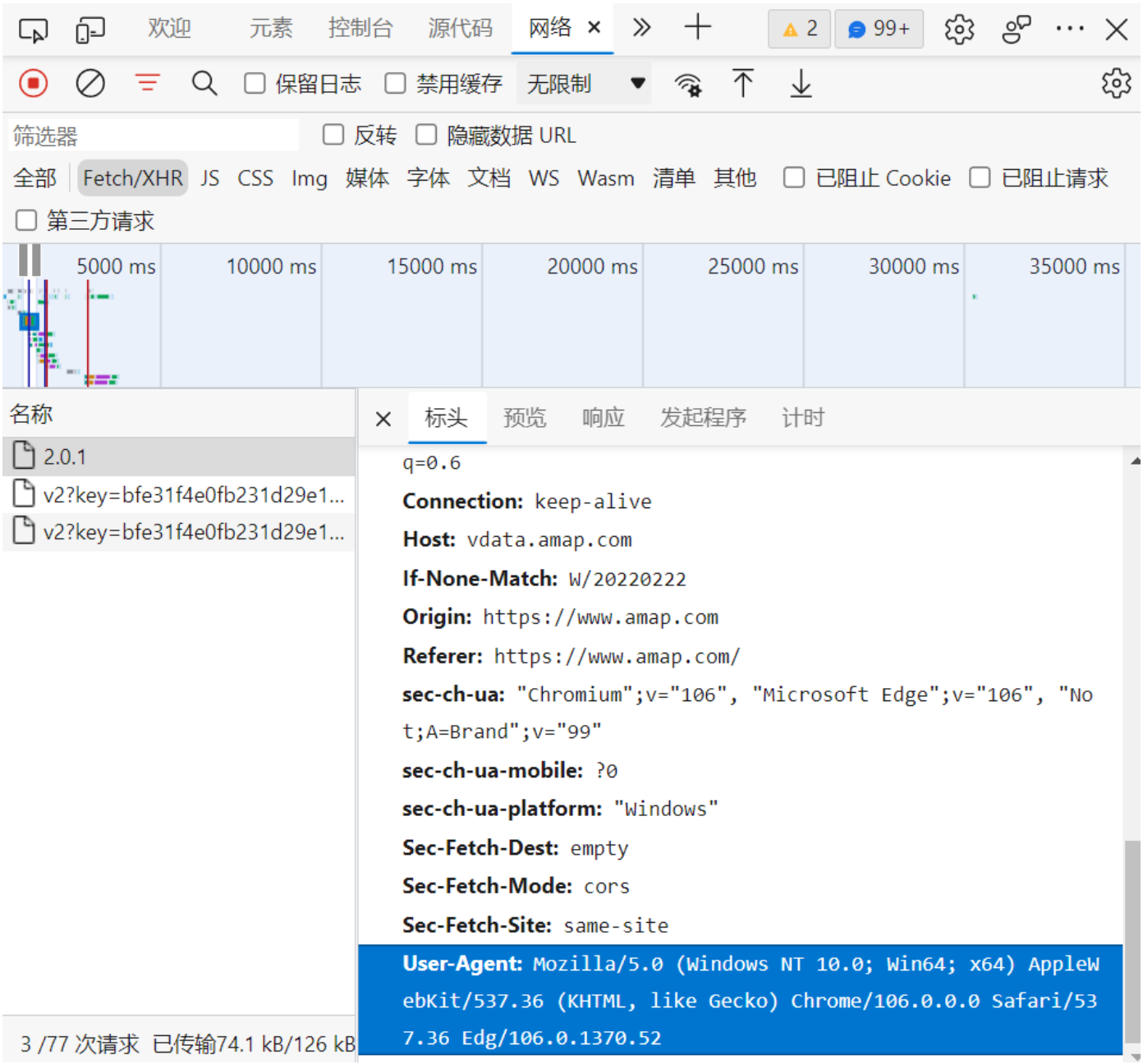
In [13]:

```
import requests
import json
import time

start = time.process_time() #程序开始计时

# def get_location(county): # 设置函数转换计算经纬度
# def get_distance(origin,destination):#设置函数计算两经纬度间驾车距离
result=int(get_distance(get_location('北京工业大学'),get_location('北京工业大学通州校区')))/1000  #

end = time.process_time() #程序结束计时

duration=end-start #程序运行所需时间

print('两地距离为:'+str(result)+'公里') #显示两地间距离
print('计算耗时:'+str(duration)+'秒') #显示程序运行所需时间
```

两地距离为:22.264公里
计算耗时:0.109375秒

### 看一下北工大的图片

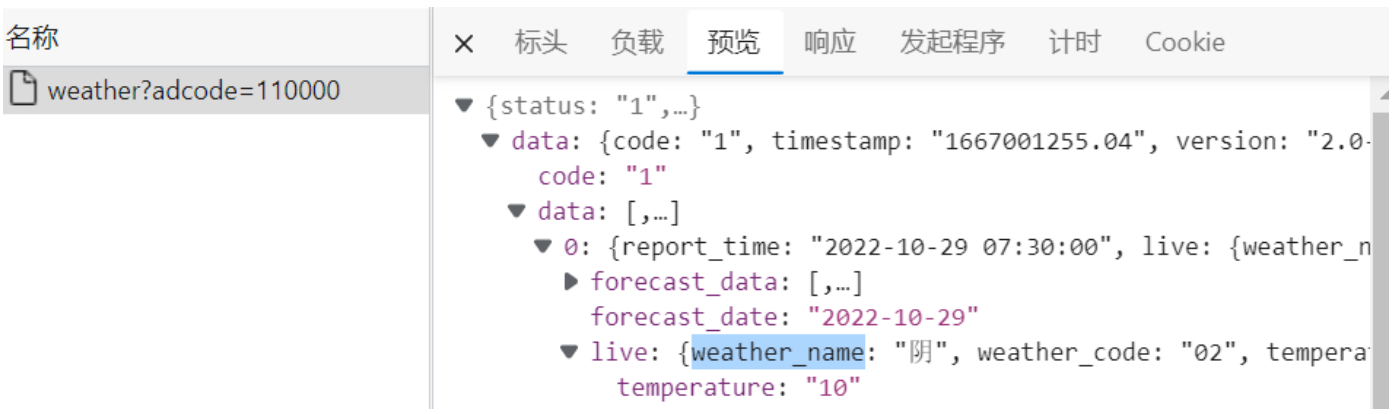

然后进行爬虫，第一步是找我的user-agent




复制一下User-Agent的值

Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36 Edg/106.0.1370.52

XHR分类找到下面数据



得到对应的url的值为

https://www.amap.com/service/weather?adcode=110000 (https://www.amap.com/service/weather?adcode=110000)

各城市对应code的url：

https://www.amap.com/service/cityList?version=20207 (https://www.amap.com/service/cityList?version=20207)

{"status":"1","data":{"cityData":{"cities":{},"hotCitys":[{"adcode":"100000","name":"全国","label":"全国","spell":"","x":"116.3683244","y":"39.915085","citycode":"total"},
{"short":"BJ","areacode":"010","adcode":"110000","name_en":"Beijing","name":"北京","x":"116.407387","y":"39.904179","label":"北京市","spell":"BeiJingShi","cities":[],"index":"B"},
{"short":"TJ","areacode":"022","adcode":"120000","name_en":"Tianjin","name":"天津","x":"117.201509","y":"39.085318","label":"天津市","spell":"TianJinShi","cities":[],"index":"T"},
{"short":"SY","areacode":"024","adcode":"210100","name_en":"Shenyang","name":"沈阳","x":"123.464675","y":"41.677576","label":"沈阳市","spell":"ShenYangShi","cities":[],"index":"S"},

查询所有城市的名称和编号

In [15]:

```
url_city = "https://www.amap.com/service/cityList?version=20207"

headers = {
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
}

city = []
response = requests.get(url=url_city, headers=headers)
content = response.json()
print(content)
```

{'status': '1', 'data': {'cityData': {'cities': {}, 'hotCitys': [{'adcode': '1000
00', 'name': '全国', 'label': '全国', 'spell': '', 'x': '116.3683244', 'y': '39.9
15085', 'citycode': 'total'}, {'short': 'BJ', 'areacode': '010', 'adcode': '11000
0', 'name_en': 'Beijing', 'name': '北京', 'x': '116.407387', 'y': '39.904179', 'l
abel': '北京市', 'spell': 'BeiJingShi', 'cities': [], 'index': 'B'}, {'short': 'T
J', 'areacode': '022', 'adcode': '120000', 'name_en': 'Tianjin', 'name': '天津',
'x': '117.201509', 'y': '39.085318', 'label': '天津市', 'spell': 'TianJinShi', 'c
ities': [], 'index': 'T'}, {'short': 'SY', 'areacode': '024', 'adcode': '210100',
'name_en': 'Shenyang', 'name': '沈阳', 'x': '123.464675', 'y': '41.677576', 'labe
l': '沈阳市', 'spell': 'ShenYangShi', 'cities': [], 'index': 'S'}, {'short': 'D
L', 'areacode': '0411', 'adcode': '210200', 'name_en': 'Dalian', 'name': '大连',
'x': '121.614786', 'y': '38.913962', 'label': '大连市', 'spell': 'DaLianShi', 'ci
ties': [], 'index': 'D'}, {'short': 'SH', 'areacode': '021', 'adcode': '310000',
'name_en': 'Shanghai', 'name': '上海', 'x': '121.473667', 'y': '31.230525', 'labe
l': '上海市', 'spell': 'ShangHaiShi', 'cities': [], 'index': 'S'}, {'short': 'N
J', 'areacode': '025', 'adcode': '320100', 'name_en': 'Nanjing', 'name': '南京',
'x': '118.796624', 'y': '32.059344', 'label': '南京市', 'spell': 'NanJingShi', 'c
ities': [], 'index': 'N'}, {'short': 'SZ', 'areacode': '0512', 'adcode': '32050
0', 'name_en': 'Suzhou', 'name': '苏州', 'x': '120.585294', 'y': '31.299758', 'la

In [18]:

```
def get_city():
    """查询所有城市名称和编号"""
    city = []
    response = requests.get(url=url_city, headers=headers)
    content = response.json()

    if "data" in content:
        cityByLetter = content["data"]["cityByLetter"]
        for k, v in cityByLetter.items():
            city.extend(v)
    return city
```

In [20]:

```
def get_weather(adcode, name):
    """根据编号查询天气"""
    item = {}
    item["adcode"] = str(adcode)
    item["name"] = name

    response = requests.get(url=url_weather.format(adcode), headers=headers)
    content = response.json()
    item["weather_name"] = content["data"]["data"][0]["forecast_data"][0]["weather_name"]
    item["min_temp"] = content["data"]["data"][0]["forecast_data"][0]["min_temp"]
    item["max_temp"] = content["data"]["data"][0]["forecast_data"][0]["max_temp"]
    return item
```

In [21]:

```
def save(item):
    """保存"""
    print(item)
    with open("weather.txt","a",encoding="utf-8") as file:
        file.write(",".join(item.values()))
        file.write("\n")
```

In [22]:

```
if __name__ == '__main__':
    city_list = get_city()
    for city in city_list:
        item = get_weather(city["adcode"],city["name"])
        save(item)
```

```
{'adcode': '152900', 'name': '阿拉善盟', 'weather_name': '阴', 'min_temp': '9',
'max_temp': '14'}
{'adcode': '210300', 'name': '鞍山', 'weather_name': '阴', 'min_temp': '9', 'max_
temp': '14'}
{'adcode': '340800', 'name': '安庆', 'weather_name': '阴', 'min_temp': '9', 'max_
temp': '14'}
{'adcode': '410500', 'name': '安阳', 'weather_name': '阴', 'min_temp': '9', 'max_
temp': '14'}
{'adcode': '513200', 'name': '阿坝藏族羌族自治州', 'weather_name': '阴', 'min_tem
p': '9', 'max_temp': '14'}
{'adcode': '520400', 'name': '安顺', 'weather_name': '阴', 'min_temp': '9', 'max_
temp': '14'}
{'adcode': '542500', 'name': '阿里', 'weather_name': '阴', 'min_temp': '9', 'max_
temp': '14'}
{'adcode': '610900', 'name': '安康', 'weather_name': '阴', 'min_temp': '9', 'max_
temp': '14'}
{'adcode': '652900', 'name': '阿克苏', 'weather_name': '阴', 'min_temp': '9', 'ma
x_temp': '14'}
{'adcode': '654300', 'name': '阿勒泰', 'weather_name': '阴', 'min_temp': '9', 'ma
```

In [26]:

```
from urllib.parse import quote

from urllib import request

import json
```


北足球场
游泳馆
材料科学
与工程学院
中国石油
祁家坟公共
汽车站
京工业大学
知新园
第四教学楼
116.491255,39.8812
第二田径场
艺术设计学院
城市综合治理
分指挥部
出口
北京工业大学
计算机学院
美食园
经管楼
篮球场
北京工业大学
学生宿舍
城建楼
职工之家
实训楼
奥林匹克体育馆
北京工大
建国饭店
人文楼

In [ ]: