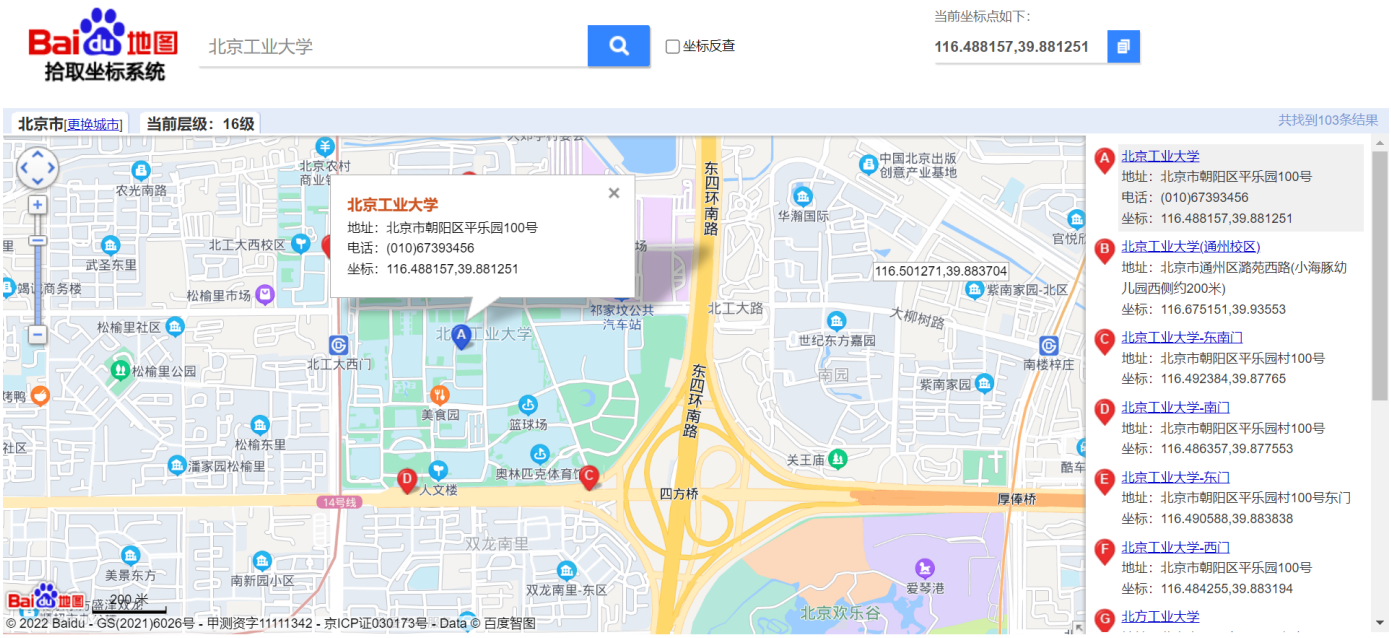
key:百度地图api

wBWkRsIdxqGVlm6GrugVjgOTXM6lpQk1

使用百度地图api：

使用百度地图api的接口，查询不同地点之间的距离，获取经纬度位置，搜索学校内已有建筑物的位置，然后进行计算， 得到两个点之间的距离，换算成步行时间，帮助用户更好的了解从一个教学楼或到一个餐厅/宿舍所用时间和距离。

用开发里面的拾取坐标系统



北京工业大学-东南门： 116.492384,39.87765

北京工业大学-南门： 116.486357,39.877553

北京工业大学-东门：116.490588,39.883838

北京工业大学-西门： 116.484255,39.883194

北京工业大学-北门： 116.488396,39.885132

北京工业大学耿丹学院：116.658646,40.221871

北京工业大学（通州校区）： 116.675151,39.93553

北京工业大学美食园： 116.487359,39.879734

北京工业大学医院： 116.488665,39.885073

北京工业大学计算机学院： 116.484856,39.880495

北京工业大学奥林匹克体育馆： 116.490691,39.878221

北京工业大学实训楼： 116.492069,39.878519

北京工业大学艺术设计学院： 116.491641,39.880523

北京工业大学第四教学楼： 116.490934,39.881173

北京工业大学游泳馆： 116.488215,39.882659

北京工业大学篮球场： 116.490288,39.879483

北京工业大学经管楼： 116.485135,39.879311

北京工业大学北足球场： 116.488828,39.883067

北京工大建国饭店： 116.484862,39.877948

为了方便对数据进行处理，我们把以上包括地点信息和经纬度信息放到excel表里，方便python后续读取数据

In [5]:

```
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 [6]:

```
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 [7]:

```
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:0.140625秒

In [8]:

```
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('北京工业大学耿丹学院')))/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:54.852公里
计算耗时:0.125秒

使用百度地图的批量算路服务，也就是我们输入经纬度信息然后他会帮我们自动用横纵坐标计算距离

实例url

https://api.map.baidu.com/routematrix/v2/driving?output=json&origins=40.45,116.34|40.54,116.35&destinations=40.34,116.45|40.35,116.46&ak=您的AK (https://api.map.baidu.com/routematrix/v2/driving?output=json&origins=40.45,116.34%7C40.54,116.35&destinations=40.34,116.45%7C40.35,116.46&ak=%E6%82%

//GET请求

步行:

https://api.map.baidu.com/routematrix/v2/walking (https://api.map.baidu.com/routematrix/v2/walking)? //GET请求

这里用步行随便测试两个点，比如我选择的是

北京工业大学实训楼： 116.492069,39.878519

北京工业大学艺术设计学院： 116.491641,39.880523

ak值： wBWkRsIdxqGVlm6GrugVjgOTXM6lpQk1

In [12]:

```
import requests
import lxml
```

In [16]:

```
url = "https://api.map.baidu.com/routematrix/v2/walking?output=json&origins=39.878519|116.492069&des

headers = {
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
}
response = requests.get(url=url,headers=headers)
content = response.text
decodejson = json.loads(content)
decodejson
```

Out[16]:

```
{'status': 2, 'message': '参数错误'}
```

In [18]:

```
!pip install geopy
```

```
Collecting geopy
  Downloading geopy-2.2.0-py3-none-any.whl (118 kB)
     ---------------------------------- 118.9/118.9 kB 434.5 kB/s eta 0:00:00
Collecting geographiclib<2,>=1.49
  Downloading geographiclib-1.52-py3-none-any.whl (38 kB)
Installing collected packages: geographiclib, geopy
Successfully installed geographiclib-1.52 geopy-2.2.0


[notice] A new release of pip available: 22.2.2 -> 22.3
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [21]:

```
from geopy.distance import geodesic

aa = geodesic((39.878519,116.492069), (39.880523,116.491641)).m
```

In [22]:

```
aa
```

Out[22]:

225.5008697570725

In [28]:

```
!pip3 install openpyxl
```

```
Requirement already satisfied: openpyxl in d:\anaconda\lib\site-packages (3.0.9)
Requirement already satisfied: et-xmlfile in d:\anaconda\lib\site-packages (from ope
npyxl) (1.1.0)


[notice] A new release of pip available: 22.2.2 -> 22.3
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [34]:

```
import pandas as pd
#读取第一列、第二列、第四列
df = pd.read_excel('pos.xlsx',usecols=[0,1,2])
data = df.values
print(data)
```

```
[['北京工业大学-东南门' 39.87765 116.492384]
 ['北京工业大学-南门' 39.877553 116.486357]
 ['北京工业大学-东门' 39.883838 116.490588]
 ['北京工业大学-西门' 39.883194 116.484255]
 ['北京工业大学-北门' 39.885132 116.488396]
 ['北京工业大学耿丹学院' 40.221871 116.658646]
 ['北京工业大学（通州校区）' 39.93553 116.675151]
 ['北京工业大学美食园' 39.879734 116.487359]
 ['北京工业大学医院' 39.885073 116.488665]
 ['北京工业大学计算机学院' 39.880495 116.484856]
 ['北京工业大学奥林匹克体育馆' 39.878221 116.490691]
 ['北京工业大学实训楼' 39.878519 116.492069]
 ['北京工业大学艺术设计学院' 39.880523 116.491641]
 ['北京工业大学第四教学楼' 39.881173 116.490934]
 ['北京工业大学游泳馆' 39.882659 116.488215]
 ['北京工业大学篮球场' 39.879483 116.490288]
 ['北京工业大学经管楼' 39.879311 116.485135]
 ['北京工业大学北足球场' 39.883067 116.488828]
 ['北京工大建国饭店' 39.877948 116.484862]]
```

上面一共有19个地点

测试一个demo，用第一个地点：北京工业大学-东南门为例

In [37]:

```
item = []
for i in range(0,19):
    item.append(geodesic((data[i][1],data[i][2]), (39.87765,116.492384)).m)

print(item)
```

```
[0.0, 515.698998619222, 704.0354900046312, 928.6904295124781, 898.0599420922853, 407
68.97989217954, 16898.178779487105, 488.18408791460195, 883.4592364858678, 717.28067
78414384, 158.09774573150912, 100.17926498077149, 325.2662523044458, 410.36213902372
293, 660.6806675554814, 271.2389312708766, 646.9592512676369, 674.0090438496496, 64
4.3262625026008]
```

然后我们使用所有的地点，也就是生成19*19的二位列表

In [48]:

```
matrix = []
item1 = []
item2 = []
item3 = []
item4 = []
item5 = []
item6 = []
item7 = []
item8 = []
item9 = []
item10 = []
item11 = []
item12 = []
item13 = []
item14 = []
item15 = []
item16 = []
item17 = []
item18 = []
item19 = []

for i in range(0,19):
        item1.append(geodesic((data[i][1],data[i][2]), (data[0][1],data[0][2])).m)
        item2.append(geodesic((data[i][1],data[i][2]), (data[1][1],data[1][2])).m)
        item3.append(geodesic((data[i][1],data[i][2]), (data[2][1],data[2][2])).m)
        item4.append(geodesic((data[i][1],data[i][2]), (data[3][1],data[3][2])).m)
        item5.append(geodesic((data[i][1],data[i][2]), (data[4][1],data[4][2])).m)
        item6.append(geodesic((data[i][1],data[i][2]), (data[5][1],data[5][2])).m)
        item7.append(geodesic((data[i][1],data[i][2]), (data[6][1],data[6][2])).m)
        item8.append(geodesic((data[i][1],data[i][2]), (data[7][1],data[7][2])).m)
        item9.append(geodesic((data[i][1],data[i][2]), (data[8][1],data[8][2])).m)
        item10.append(geodesic((data[i][1],data[i][2]), (data[9][1],data[9][2])).m)
        item11.append(geodesic((data[i][1],data[i][2]), (data[10][1],data[10][2])).m)
        item12.append(geodesic((data[i][1],data[i][2]), (data[11][1],data[11][2])).m)
        item13.append(geodesic((data[i][1],data[i][2]), (data[12][1],data[12][2])).m)
        item14.append(geodesic((data[i][1],data[i][2]), (data[13][1],data[13][2])).m)
        item15.append(geodesic((data[i][1],data[i][2]), (data[14][1],data[14][2])).m)
        item16.append(geodesic((data[i][1],data[i][2]), (data[15][1],data[15][2])).m)
        item17.append(geodesic((data[i][1],data[i][2]), (data[16][1],data[16][2])).m)
        item18.append(geodesic((data[i][1],data[i][2]), (data[17][1],data[17][2])).m)
        item19.append(geodesic((data[i][1],data[i][2]), (data[18][1],data[18][2])).m)
matrix.append(item1)
matrix.append(item2)
matrix.append(item3)
matrix.append(item4)
matrix.append(item5)
matrix.append(item6)
matrix.append(item7)
matrix.append(item8)
matrix.append(item9)
matrix.append(item10)
matrix.append(item11)
matrix.append(item12)
matrix.append(item13)
matrix.append(item14)
matrix.append(item15)
matrix.append(item16)
matrix.append(item17)
matrix.append(item18)
```

```
matrix.append(item19)


print(matrix)
```

```
[[0.0, 515.698998619222, 704.0354900046312, 928.6904295124781, 898.0599420922853, 40
768.97989217954, 16898.178779487105, 488.18408791460195, 883.4592364858678, 717.2806
778414384, 158.09774573150912, 100.17926498077149, 325.2662523044458, 410.3621390237
2293, 660.6806675554814, 271.2389312708766, 646.9592512676369, 674.0090438496496, 64
4.3262625026008], [515.698998619222, 0.0, 786.1114014412678, 651.6328901953766, 859.
3998907112943, 40960.845209567735, 17379.924398012517, 256.88399560452126, 857.98731
53401323, 350.9870733874408, 378.1014176555256, 500.26957113780924, 559.521525121599
9, 561.117360795122, 588.7889478648646, 398.7526392813035, 221.42447860635846, 647.6
94571004592, 135.2022857212466], [704.0354900046312, 786.1114014412678, 0.0, 546.415
919655161, 236.2168797665156, 40179.78046831886, 16792.587746627847, 532.85480208926
31, 214.14912773488777, 614.9692138973892, 623.7309278593925, 604.0162709701199, 37
8.93345263581836, 297.37777593697, 241.53534669331017, 484.22641011781565, 685.73448
71198177, 173.18545397462756, 817.0728152238703], [928.6904295124781, 651.6328901953
766, 546.415919655161, 0.0, 414.4507912928241, 40442.30095463833, 17326.39957407963
4, 467.0001187682575, 431.0723258313089, 304.0541183485502, 779.7372985340611, 846.3
056352193294, 697.9465274985004, 613.8122839361931, 343.9055582401857, 660.384679295
7962, 437.66091557825547, 391.42511228824935, 584.7854378304182], [898.059942092285
3, 859.3998907112943, 236.2168797665156, 414.4507912928241, 0.0, 40112.90650291054,
16920.652493630583, 605.8814423566573, 23.9237752879178, 597.3039504943314, 792.0593
772438011, 798.6549675852349, 582.181045690329, 490.26515801055245, 275.019274594860
1, 647.7655235420793, 703.946265944946, 232.2405671484158, 853.0197855944408], [4076
8.97989217954, 40960.845209567735, 40179.78046831886, 40442.30095463833, 40112.90650
291054, 0.0, 31825.353476573637, 40704.04825708797, 40110.70964536511, 40702.4983731
61376, 40760.09985016266, 40687.88851331065, 40492.23949155715, 40445.97574809705, 4
0374.62787199689, 40641.05095605695, 40816.35751817337, 40313.524705789554, 40965.90
38082844], [16898.178779487105, 17379.924398012517, 16792.587746627847, 17326.399574
079634, 16920.652493630583, 31825.353476573637, 0.0, 17211.490044396345, 16901.12428
3821482, 17381.373789782345, 17008.27687103116, 16886.596657952665, 16838.3410059747
44, 16868.666814529748, 17028.230010876196, 16988.277287558067, 17405.884437424847,
16963.38340394363, 17482.60370840424], [488.18408791460195, 256.88399560452126, 532.
8548020892631, 467.0001187682575, 605.8814423566573, 40704.04825708797, 17211.490044
396345, 0.0, 603.2365760440507, 230.18335000451972, 330.8559462525988, 424.898013842
25973, 376.6253264442294, 345.0375026106875, 332.92193389413836, 252.1024642906738,
195.96085834738668, 390.8234163894521, 291.4642065758923], [883.4592364858678, 857.9
873153401323, 214.14912773488777, 431.0723258313089, 23.9237752879178, 40110.7096453
6511, 16901.124283821482, 603.2365760440507, 0.0, 603.7667992458655, 780.28338130675
41, 783.8001705433187, 565.7103746805399, 474.533634916311, 270.78205881602844, 636.
008443940479, 707.4476686966663, 223.16697776600154, 855.381096678009], [717.2806778
414384, 350.9870733874408, 614.9692138973892, 304.0541183485502, 597.3039504943314,
40702.498373161376, 17381.373789782345, 230.18335000451972, 603.7667992458655, 0.0,
559.3742793998907, 654.8731229529143, 580.4143026457567, 525.3465850065554, 374.5554
3778275825, 478.0632452499679, 133.61116199481074, 443.8422185606413, 282.7997655912
2713], [158.09774573150912, 378.1014176555256, 623.7309278593925, 779.7372985340611,
792.0593772438011, 40760.09985016266, 17008.27687103116, 330.8559462525988, 780.2833
813067541, 559.3742793998907, 0.0, 122.43684128882406, 268.20476588388317, 328.42591
217880937, 536.3530721557933, 144.30134843872244, 490.4529812885095, 561.16737124379
04, 499.56534364160933], [100.17926498077149, 500.26957113780924, 604.0162709701199,
846.3056352193294, 798.6549675852349, 40687.88851331065, 16886.596657952665, 424.898
01384225973, 783.8001705433187, 654.8731229529143, 122.43684128882406, 0.0, 225.5008
697570725, 310.26275100368224, 565.6759625178236, 186.19461351674707, 599.6485492745
12, 576.0755655781145, 619.7765149400416], [325.2662523044458, 559.5215251215999, 37
8.93345263581836, 697.9465274985004, 582.181045690329, 40492.23949155715, 16838.3410
05974744, 376.6253264442294, 565.7103746805399, 580.4143026457567, 268.2047658838831
7, 225.5008697570725, 0.0, 94.16090877369277, 377.0064106741522, 163.49273964647574,
572.5828073080078, 371.06386200380484, 646.5535637453836], [410.36213902372293, 561.
117360795122, 297.37777593697, 613.8122839361931, 490.26515801055245, 40445.97574809
```

```
705, 16868.666814529748, 345.0375026106875, 474.533634916311, 525.3465850065554, 32
8.42591217880937, 310.26275100368224, 94.16090877369277, 0.0, 285.1648569189826, 19
5.612493330273, 537.4205215731357, 276.9070859589468, 630.8876599075912], [660.68066
75554814, 588.7889478648646, 241.53534669331017, 343.9055582401857, 275.019274594860
1, 40374.62787199689, 17028.230010876196, 332.92193389413836, 270.78205881602844, 37
4.55543778275825, 536.3530721557933, 565.6759625178236, 377.0064106741522, 285.16485
69189826, 0.0, 394.71428373199626, 455.63566125011346, 69.29437033334926, 596.551952
2591641], [271.2389312708766, 398.7526392813035, 484.22641011781565, 660.38467929579
62, 647.7655235420793, 40641.05095605695, 16988.277287558067, 252.1024642906738, 63
6.008443940479, 478.0632452499679, 144.30134843872244, 186.19461351674707, 163.49273
964647574, 195.612493330273, 394.71428373199626, 0.0, 441.22120639345576, 417.077794
0568004, 494.46721443694696], [646.9592512676369, 221.42447860635846, 685.7344871198
177, 437.66091557825547, 703.946265944946, 40816.35751817337, 17405.884437424847, 19
5.96085834738668, 707.4476686966663, 133.61116199481074, 490.4529812885095, 599.6485
49274512, 572.5828073080078, 537.4205215731357, 455.63566125011346, 441.221206393455
76, 0.0, 523.1792241728672, 153.1283544659409], [674.0090438496496, 647.69457100459
2, 173.18545397462756, 391.42511228824935, 232.2405671484158, 40313.524705789554, 16
963.38340394363, 390.8234163894521, 223.16697776600154, 443.8422185606413, 561.16737
12437904, 576.0755655781145, 371.06386200380484, 276.9070859589468, 69.2943703333492
6, 417.0777940568004, 523.1792241728672, 0.0, 661.9275363803721], [644.326262502600
8, 135.2022857212466, 817.0728152238703, 584.7854378304182, 853.0197855944408, 4096
5.9038082844, 17482.60370840424, 291.4642065758923, 855.381096678009, 282.7997655912
2713, 499.56534364160933, 619.7765149400416, 646.5535637453836, 630.8876599075912, 5
96.5519522591641, 494.46721443694696, 153.1283544659409, 661.9275363803721, 0.0]]
```

因为已经爬取了数据，下一步的内容就是将数据导出到另一个excel文件里面

In [80]:

```
from openpyxl import Workbook
# 创建一个Workbook对象
wb = Workbook()
# 获取当前活跃的sheet，默认是第一个sheet
ws = wb.active

ws['A1'] = 'place_start'
ws['B1'] = 'place_end'
ws['C1'].value = 'distance'
# row1 = ['class1', 'zhangsan', 90]
# row2 = ['class2', 'lisi', 88]
column1 = ['北京工业大学-东南门']
column2 = ['北京工业大学-东南门','北京工业大学-南门','北京工业大学-东门','北京工业大学-西门','北京工
           '北京工业大学耿丹学院','北京工业大学（通州校区）','北京工业大学美食园','北京工业大学医院'
           '北京工业大学奥林匹克体育馆','北京工业大学实训楼','北京工业大学艺术设计学院','北京工业大学
           '北京工业大学游泳馆','北京工业大学篮球场','北京工业大学经管楼','北京工业大学北足球场','北
            ]
#for i in range(0,19):
#    column3 = places[n]


# places = [place0,place1,place2,place3,place4,place5,place6,place7,place8,place9,place10,place11,pl
place0  = [0.0, 515.698998619222, 704.0354900046312, 928.6904295124781, 898.0599420922853,
           40768.97989217954, 16898.178779487105, 488.18408791460195, 883.4592364858678,
           717.2806778414384, 158.09774573150912, 100.17926498077149, 325.2662523044458,
           410.36213902372293, 660.6806675554814, 271.2389312708766, 646.9592512676369,
           674.0090438496496, 644.3262625026008]

place1 = [515.698998619222, 0.0, 786.1114014412678, 651.6328901953766, 859.3998907112943,
          40960.845209567735, 17379.924398012517, 256.88399560452126, 857.9873153401323,
          350.9870733874408, 378.1014176555256, 500.26957113780924, 559.5215251215999,
          561.117360795122, 588.7889478648646, 398.7526392813035, 221.42447860635846,
          647.694571004592, 135.2022857212466]

place2 = [704.0354900046312, 786.1114014412678, 0.0, 546.415919655161, 236.2168797665156, 40179.7804
place3 = [928.6904295124781, 651.6328901953766, 546.415919655161, 0.0, 414.4507912928241, 40442.3009
place4 = [898.0599420922853, 859.3998907112943, 236.2168797665156, 414.4507912928241, 0.0, 40112.906
place5 = [40768.97989217954, 40960.845209567735, 40179.78046831886, 40442.30095463833, 40112.9065029
place6 = [16898.178779487105, 17379.924398012517, 16792.587746627847, 17326.399574079634, 16920.6524
place7 = [488.18408791460195, 256.88399560452126, 532.8548020892631, 467.0001187682575, 605.88144235
place8 = [883.4592364858678, 857.9873153401323, 214.14912773488777, 431.0723258313089, 23.9237752879
place9 = [717.2806778414384, 350.9870733874408, 614.9692138973892, 304.0541183485502, 597.3039504943
place10 = [158.09774573150912, 378.1014176555256, 623.7309278593925, 779.7372985340611, 792.05937724
place11 = [100.17926498077149, 500.26957113780924, 604.0162709701199, 846.3056352193294, 798.6549675
place12 = [325.2662523044458, 559.5215251215999, 378.93345263581836, 697.9465274985004, 582.18104569
place13 = [410.36213902372293, 561.117360795122, 297.37777593697, 613.8122839361931, 490.26515801055
place14 = [660.6806675554814, 588.7889478648646, 241.53534669331017, 343.9055582401857, 275.01927459
place15 = [271.2389312708766, 398.7526392813035, 484.22641011781565, 660.3846792957962, 647.76552354
place16 = [646.9592512676369, 221.42447860635846, 685.7344871198177, 437.66091557825547, 703.9462659
place17 = [674.0090438496496, 647.694571004592, 173.18545397462756, 391.42511228824935, 232.24056714
place18 = [644.3262625026008, 135.2022857212466, 817.0728152238703, 584.7854378304182, 853.019785594


column3 = place0

ws.append(column1)
ws.append(column2)
ws.append(column3)
```

```
wb.save('data0.xlsx')
```

通过上面的程序跑出来的内容，在文件夹下，对应的分别是place0-18,也就是一共19个

所以总共的数据量一共是19*19，但是这里我们需要注意的是根据握手原理，会有一部分是是重复的。因此我们进行数据清洗，得到的结果如下。

result的是上面跑出来结果的综合，result_cleaned是已经经过数据清洗的结果。

In [81]:

```
import pandas as pd
df = pd.read_excel('result.xlsx')
result = df.values
print(result)
```

```
[['北京工业大学-东南门' '北京工业大学-东南门' 0.0]
 ['北京工业大学-东南门' '北京工业大学-南门' 515.698998619222]
 ['北京工业大学-东南门' '北京工业大学-东门' 704.0354900046312]
 ...
 ['北京工大建国饭店' '北京工业大学经管楼' 153.1283544659409]
 ['北京工大建国饭店' '北京工业大学北足球场' 661.9275363803721]
 ['北京工大建国饭店' '北京工大建国饭店' 0.0]]
```

一共172行数据经过清洗之后->清除=0的行以及=0前面的所有行

164	北京工业大学游泳馆	北京工业大学经管楼	455.0356013
165	北京工业大学游泳馆	北京工业大学北足球场	69.29437033
166	北京工业大学游泳馆	北京工大建国饭店	596.5519523
167	北京工业大学篮球场	北京工业大学经管楼	441.2212064
168	北京工业大学篮球场	北京工业大学北足球场	417.0777941
169	北京工业大学篮球场	北京工大建国饭店	494.4672144
170	北京工业大学经管楼	北京工业大学北足球场	523.1792242
171	北京工业大学经管楼	北京工大建国饭店	153.1283545
172	北京工业大学北足球场	北京工大建国饭店	661.9275364
173			

In [82]:

```
import pandas as pd
df = pd.read_excel('result_cleaned.xlsx')
result = df.values
print(result)
```

```
[['北京工业大学-东南门' '北京工业大学-南门' 515.698998619222]
 ['北京工业大学-东南门' '北京工业大学-东门' 704.0354900046312]
 ['北京工业大学-东南门' '北京工业大学-西门' 928.6904295124781]
 ['北京工业大学-东南门' '北京工业大学-北门' 898.0599420922853]
 ['北京工业大学-东南门' '北京工业大学耿丹学院' 40768.97989217954]
 ['北京工业大学-东南门' '北京工业大学（通州校区）' 16898.17877948711]
 ['北京工业大学-东南门' '北京工业大学美食园' 488.1840879146019]
 ['北京工业大学-东南门' '北京工业大学医院' 883.4592364858678]
 ['北京工业大学-东南门' '北京工业大学计算机学院' 717.2806778414384]
 ['北京工业大学-东南门' '北京工业大学奥林匹克体育馆' 158.0977457315091]
 ['北京工业大学-东南门' '北京工业大学实训楼' 100.1792649807715]
 ['北京工业大学-东南门' '北京工业大学艺术设计学院' 325.2662523044458]
 ['北京工业大学-东南门' '北京工业大学第四教学楼' 410.3621390237229]
 ['北京工业大学-东南门' '北京工业大学游泳馆' 660.6806675554814]
 ['北京工业大学-东南门' '北京工业大学篮球场' 271.2389312708766]
 ['北京工业大学-东南门' '北京工业大学经管楼' 646.9592512676369]
 ['北京工业大学-东南门' '北京工业大学北足球场' 674.0090438496496]
 ['北京工业大学-东南门' '北京工大建国饭店' 644.3262625026008]
 ['北京工业大学-南门' '北京工业大学-东门' 786.1114014412678]
```

到这里就得到了所有两点之间的通过爬虫方法的真实距离，经检验，与地图上显示一致

In [ ]: