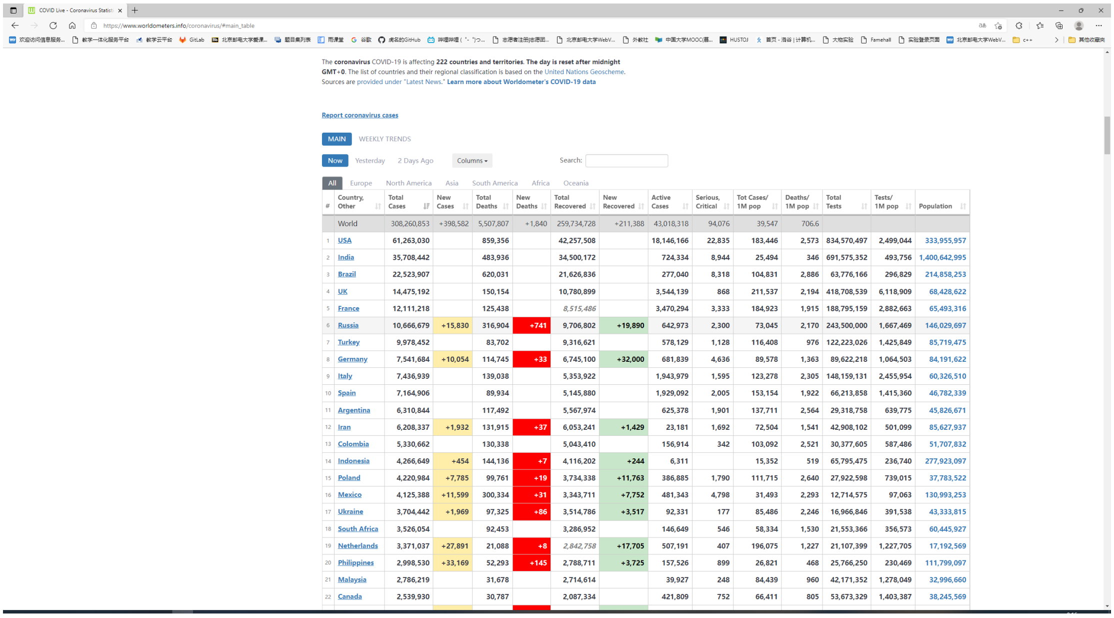
Python 程序设计：分析新冠疫情数据

一、数据来源

网址：https://www.worldometers.info/coronavirus/#main_table
首页截图：



COVID Live - Coronavirus Statisti
https://www.worldometers.info/coronavirus/#main_table

The **coronavirus** COVID-19 is affecting **222 countries and territories. The day is reset after midnight GMT+0**. The list of countries and their regional classification is based on the United Nations Geoscheme. Sources are provided under "Latest News." **Learn more about Worldometer's COVID-19 data**

Report coronavirus cases

MAIN WEEKLY TRENDS

Now Yesterday 2 Days Ago Columns ▾ Search:

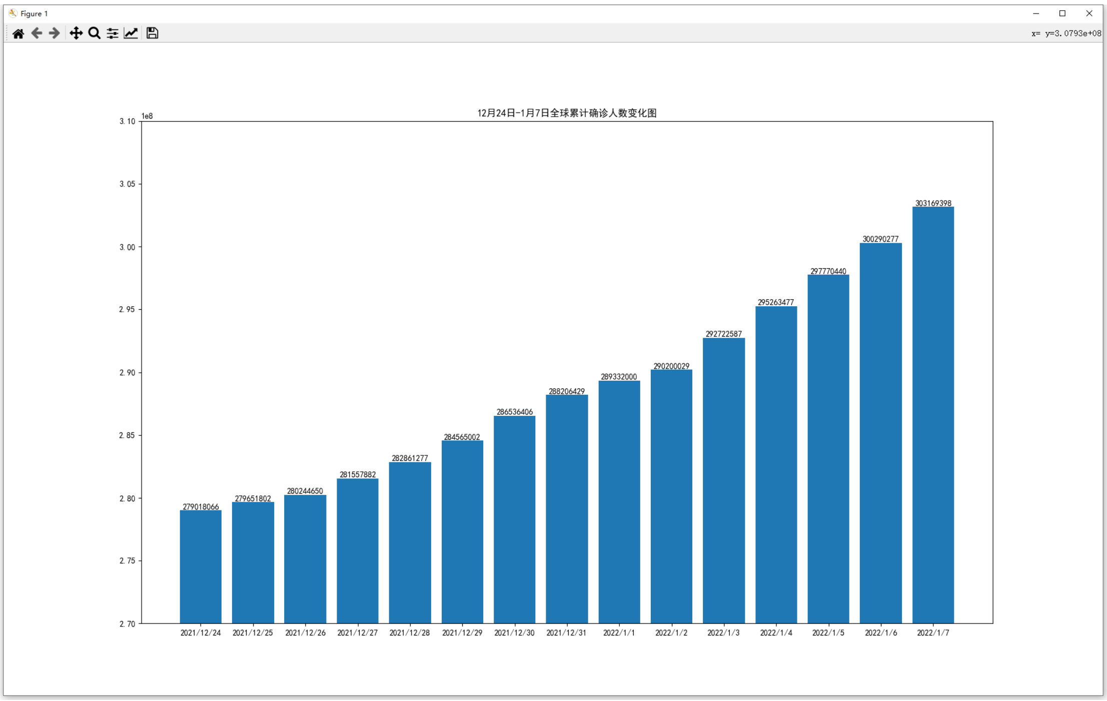
All Europe North America Asia South America Africa Oceania

#	Country, Other	Total Cases	New Cases	Total Deaths	New Deaths	Total Recovered	New Recovered	Active Cases	Serious, Critical	Tot Cases/ 1M pop	Deaths/ 1M pop	Total Tests	Tests/ 1M pop	Population
	World	308,260,853	+398,582	5,507,807	+1,840	259,734,728	+211,388	43,018,318	94,076	39,547	706.6			
1	USA	61,263,030		859,356		42,257,508		18,146,166	22,835	183,446	2,573	834,570,497	2,499,044	333,955,957
2	India	35,708,442		483,936		34,500,172		724,334	8,944	25,494	346	691,575,352	493,756	1,400,642,995
3	Brazil	22,523,907		620,031		21,626,836		277,040	8,318	104,831	2,886	63,776,166	296,829	214,858,253
4	UK	14,475,192		150,154		10,780,899		3,544,139	868	211,537	2,194	418,708,539	6,118,909	68,428,622
5	France	12,111,218		125,438		*8,515,486*		3,470,294	3,333	184,923	1,915	188,795,159	2,882,663	65,493,316
6	Russia	10,666,679	+15,830	316,904	+741	9,706,802	+19,890	642,973	2,300	73,045	2,170	243,500,000	1,667,469	146,029,697
7	Turkey	9,978,452		83,702		9,316,621		578,129	1,128	116,408	976	122,223,026	1,425,849	85,719,475
8	Germany	7,541,684	+10,054	114,745	+33	6,745,100	+32,000	681,839	4,636	89,578	1,363	89,622,218	1,064,503	84,191,622
9	Italy	7,436,939		139,038		5,353,922		1,943,979	1,595	123,278	2,305	148,159,131	2,455,954	60,326,510
10	Spain	7,164,906		89,934		5,145,880		1,929,092	2,005	153,154	1,922	66,213,858	1,415,360	46,782,339
11	Argentina	6,310,844		117,492		5,567,974		625,378	1,901	137,711	2,564	29,318,758	639,775	45,826,671
12	Iran	6,208,337	+1,932	131,915	+37	6,053,241	+1,429	23,181	1,692	72,504	1,541	42,908,102	501,099	85,627,937
13	Colombia	5,330,662		130,338		5,043,410		156,914	342	103,092	2,521	30,377,605	587,486	51,707,832
14	Indonesia	4,266,649	+454	144,136	+7	4,116,202	+244	6,311		15,352	519	65,795,475	236,740	277,923,097
15	Poland	4,220,984	+7,785	99,761	+19	3,734,338	+11,763	386,885	1,790	111,715	2,640	27,922,598	739,015	37,783,522
16	Mexico	4,125,388	+11,599	300,334	+31	3,343,711	+7,752	481,343	4,798	31,493	2,293	12,714,575	97,063	130,993,253
17	Ukraine	3,704,442	+1,969	97,325	+86	3,514,786	+3,517	92,331	177	85,486	2,246	16,966,846	391,538	43,333,815
18	South Africa	3,526,054		92,453		3,286,952		146,649	546	58,334	1,530	21,553,366	356,573	60,445,927
19	Netherlands	3,371,037	+27,891	21,088	+8	*2,842,758*	+17,705	507,191	407	196,075	1,227	21,107,399	1,227,705	17,192,569
20	Philippines	2,998,530	+33,169	52,293	+145	2,788,711	+3,725	157,526	899	26,821	468	25,766,250	230,469	111,799,097
21	Malaysia	2,786,219		31,678		2,714,614		39,927	248	84,439	960	42,171,352	1,278,049	32,996,660
22	Canada	2,539,930		30,787		2,087,334		421,809	752	66,411	805	53,673,329	1,403,387	38,245,569

二、数据分析与展示

1）　15 天中，全球新冠疫情的总体变化趋势；
代码如下：

```python
import numpy as np
import pandas as pd
import matplotlib.pylab as plt

filename = '15d_world.csv'

#导入数据：日期，累计确诊数
df = pd.read_csv(filename, encoding='utf-8', usecols=[2, 3])

X = []
Y = []
L = []
print(df)
for i in range(15):
    X.append(i + 1)
    Y.append(df.iloc[i, 1])
    L.append(df.iloc[i, 0])

plt.figure(figsize=(20, 12))
plt.rcParams['font.sans-serif'] = 'SimHei'
plt.bar(X, Y)
for a, b in zip(X, Y):
    plt.text(a, b, '%d' % b, ha='center', va='bottom')
# X 坐标轴数据
plt.xticks(X, L)
plt.ylim((2.7e8,3.1e8))
plt.title("12 月 24 日-1 月 7 日全球累计确诊人数变化图")
plt.savefig('./12 月 24 日-1 月 7 日全球累计确诊人数变化图')
plt.show()
```

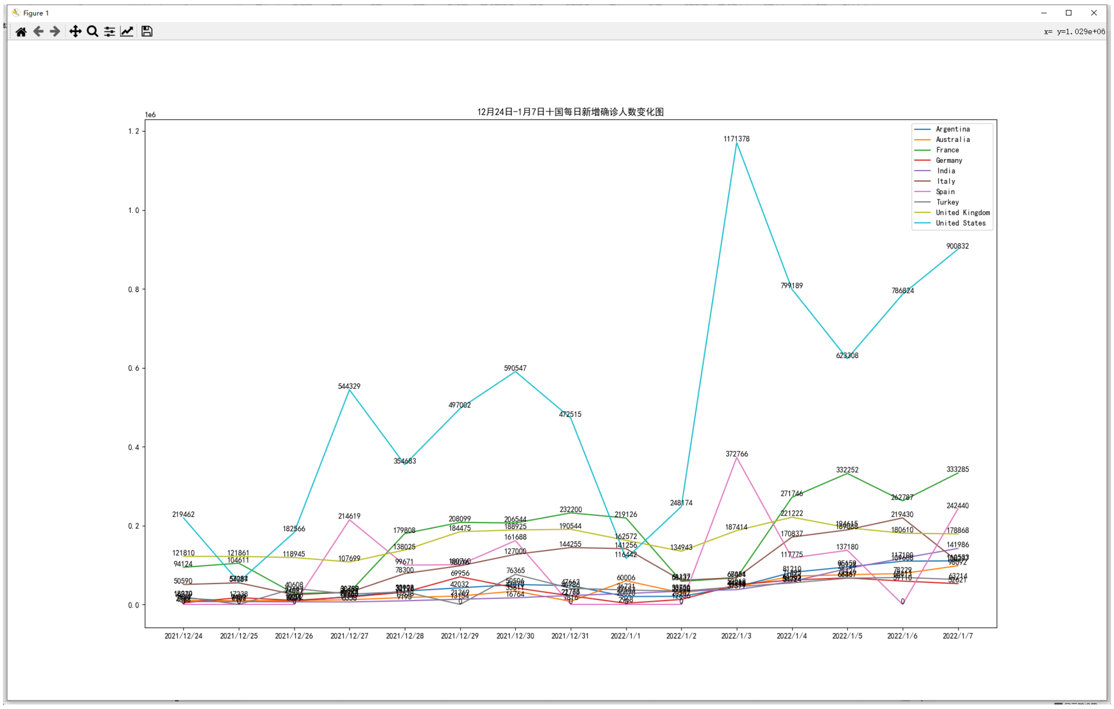


2） 15 天中，每日新增确诊数累计排名前 10 个国家的每日新增确诊数据的曲线图；

代码如下：

```
import pandas as pd
import numpy as np
import random
from matplotlib import pylab as plt

# 打开文件，读入相关数据
fileNmae = './d15_country.csv'
# 国家 日期 累计确诊 新增确诊
df = pd.read_csv(fileNmae, encoding='utf-8', usecols=[1, 2, 3, 4])

# 分析 15 天内增长最多的国家
df2 = df.copy()
```

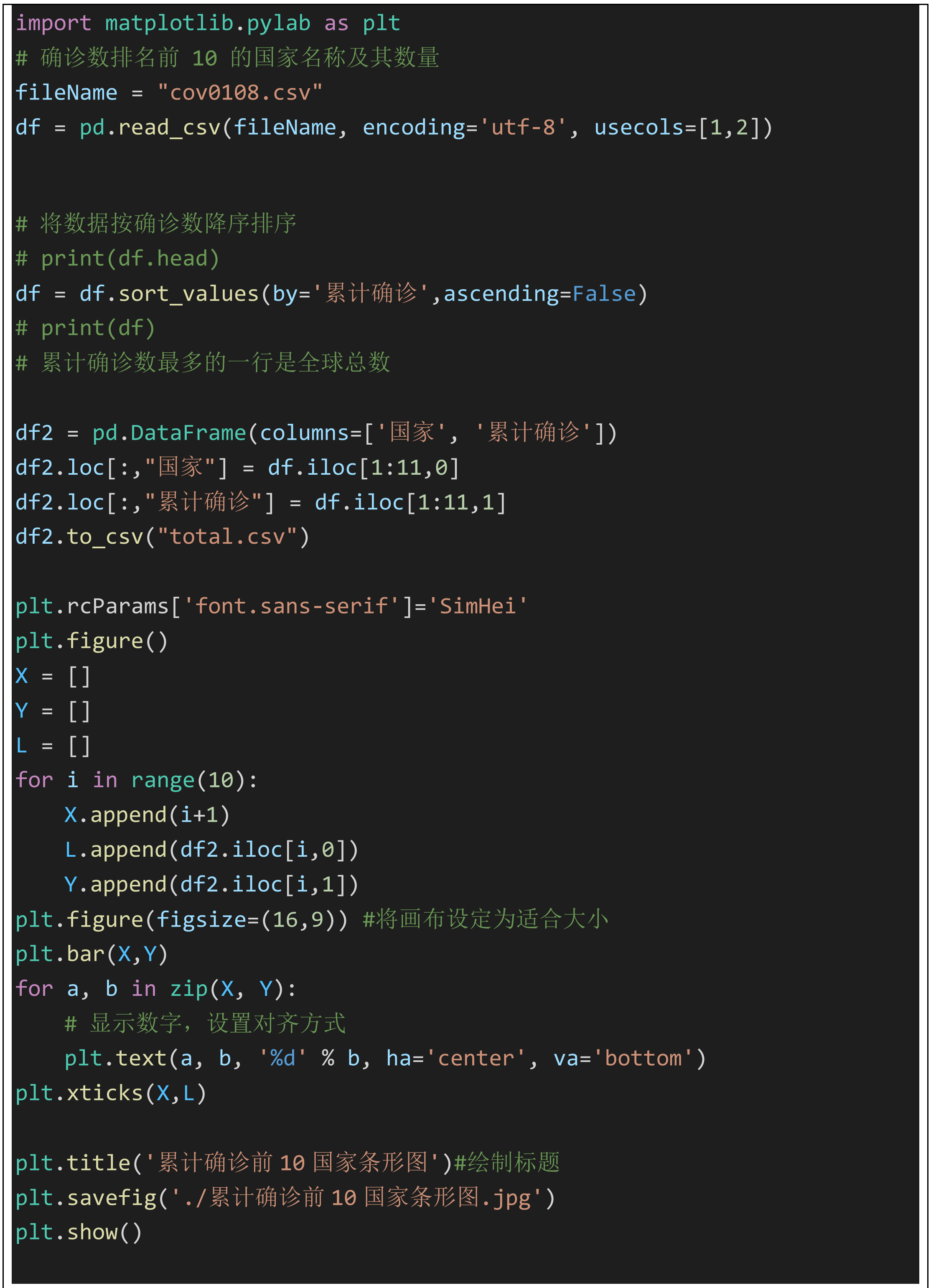
```
df2 = df2.drop(['日期', '新增确诊'], axis=1)
# df2 = df2.sort_values(by='国家')
df3 = df2.copy()
df2 = df2.groupby('国家').agg('max')
df3 = df3.groupby('国家').agg('min')
df2.to_csv('./d15max')
df3.to_csv('./d15min')
fileNmae = 'd15max'
df2 = pd.read_csv(fileNmae, encoding='utf-8', usecols=[0, 1])
fileNmae = 'd15min'
df3 = pd.read_csv(fileNmae, encoding='utf-8', usecols=[0, 1])
df4 = pd.DataFrame(columns=['国家', '15 天增长'])
for i in range(df2.shape[0]):
    df4.loc[i, '国家'] = df2.iloc[i, 0]
    df4.loc[i, '15 天增长'] = df2.iloc[i, 1] - df3.iloc[i, 1]
# print(df4)
df4 = df4.sort_values(by='15 天增长', ascending=False)
print(df4)
country = []
for i in range(10):
    country.append(df4.iloc[i, 0])
# print(country)

# 得到国家后，再筛选出相关数据

df5 = df.copy()
df6 = pd.DataFrame(columns=['国家', '日期', '新增确诊'])
j = 0
for i in range(df5.shape[0]):
    if df5.iloc[i, 0] in country:
        df6.loc[j, '国家'] = df5.iloc[i, 0]
        df6.loc[j, '日期'] = df5.iloc[i, 1]
        df6.loc[j, '新增确诊'] = df5.iloc[i, 3]
        j += 1

print(df6)
plt.figure(figsize=(20, 12))
plt.rcParams['font.sans-serif'] = 'SimHei'
X = []
L = []
C = []
for i in range(15):
    j = i * 15
    if j<150:
        C.append(df6.iloc[j, 0])
    X.append(i + 1)
    L.append(df6.iloc[i, 1])

for i in range(10):
    Y = []
    for j in range(15):
        k = j + i * 15
        Y.append(df6.iloc[k, 2])
    plt.plot(X, Y, label=C[i])
    # if C[i] in ['USA']:
    for a, b in zip(X, Y):
        plt.text(a, b, '%d' % b, ha='center', va='bottom')

plt.xticks(X, L)
plt.legend(loc='upper right')
plt.title("12 月 24 日-1 月 7 日十国每日新增确诊人数变化图")
plt.savefig('./12 月 24 日-1 月 7 日十国每日新增确诊人数变化图')
plt.show()
```



3）累计确诊数据前 10 的国家及其数量

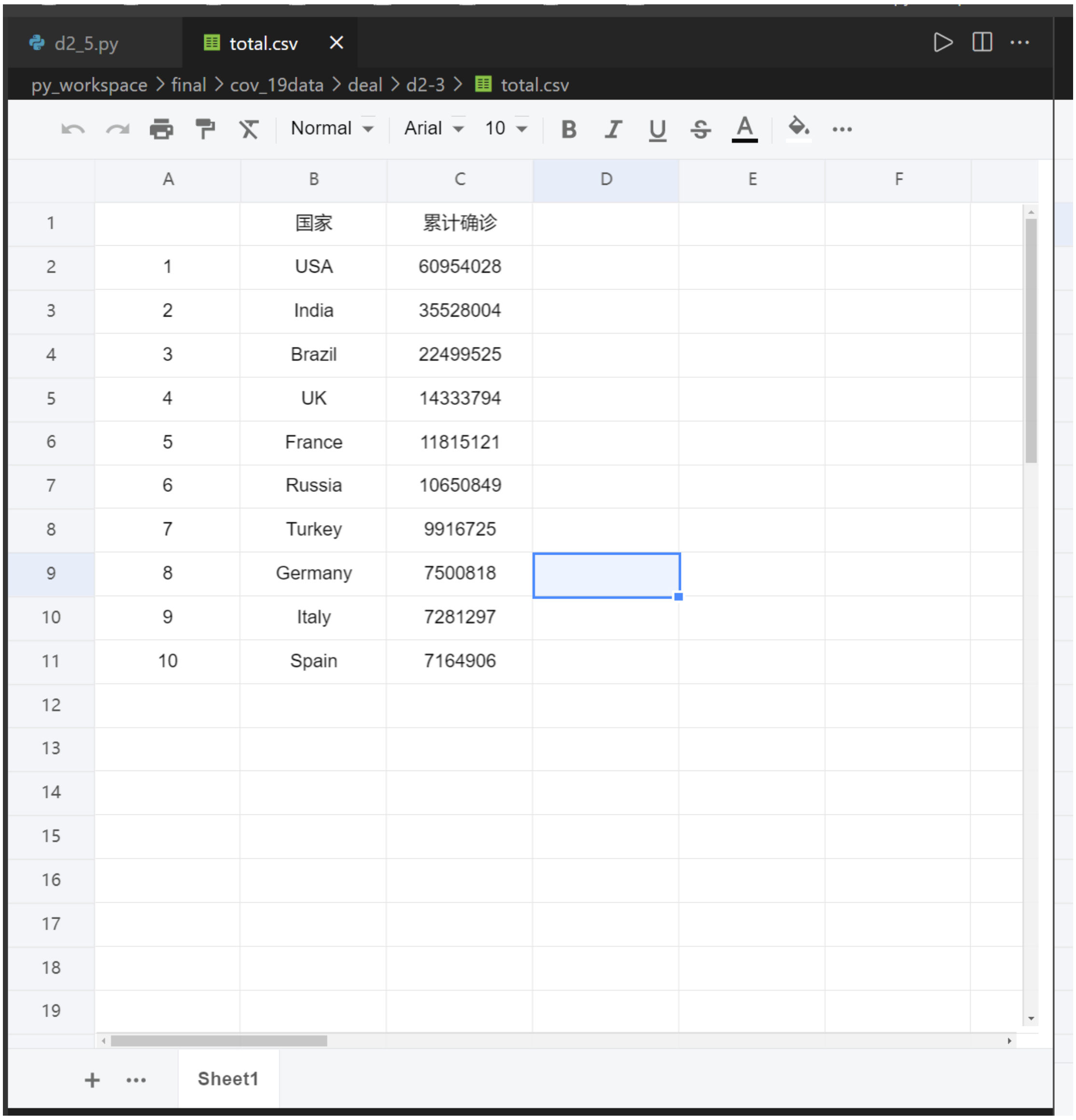
代码如下：

```
import numpy as np
import pandas as pd
import csv
```

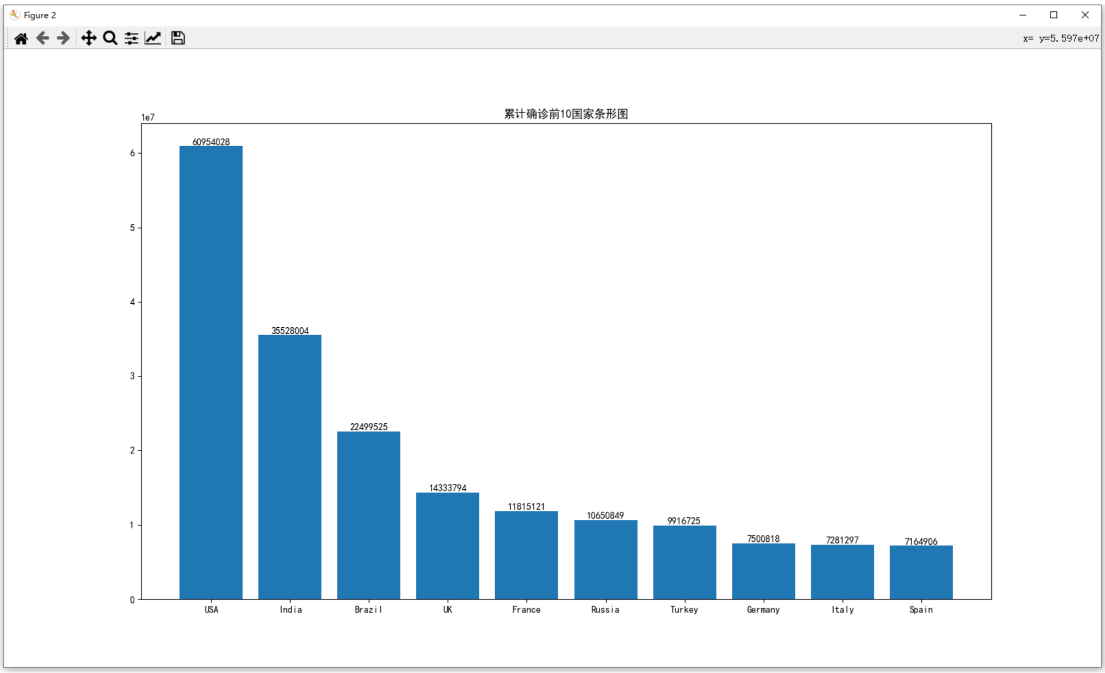


```
import matplotlib.pylab as plt
# 确诊数排名前 10 的国家名称及其数量
fileName = "cov0108.csv"
df = pd.read_csv(fileName, encoding='utf-8', usecols=[1,2])


# 将数据按确诊数降序排序
# print(df.head)
df = df.sort_values(by='累计确诊',ascending=False)
# print(df)
# 累计确诊数最多的一行是全球总数

df2 = pd.DataFrame(columns=['国家', '累计确诊'])
df2.loc[:,"国家"] = df.iloc[1:11,0]
df2.loc[:,"累计确诊"] = df.iloc[1:11,1]
df2.to_csv("total.csv")

plt.rcParams['font.sans-serif']='SimHei'
plt.figure()
X = []
Y = []
L = []
for i in range(10):
    X.append(i+1)
    L.append(df2.iloc[i,0])
    Y.append(df2.iloc[i,1])
plt.figure(figsize=(16,9)) #将画布设定为适合大小
plt.bar(X,Y)
for a, b in zip(X, Y):
    # 显示数字，设置对齐方式
    plt.text(a, b, '%d' % b, ha='center', va='bottom')
plt.xticks(X,L)

plt.title('累计确诊前 10 国家条形图')#绘制标题
plt.savefig('./累计确诊前 10 国家条形图.jpg')
plt.show()
```

结果如图所示：

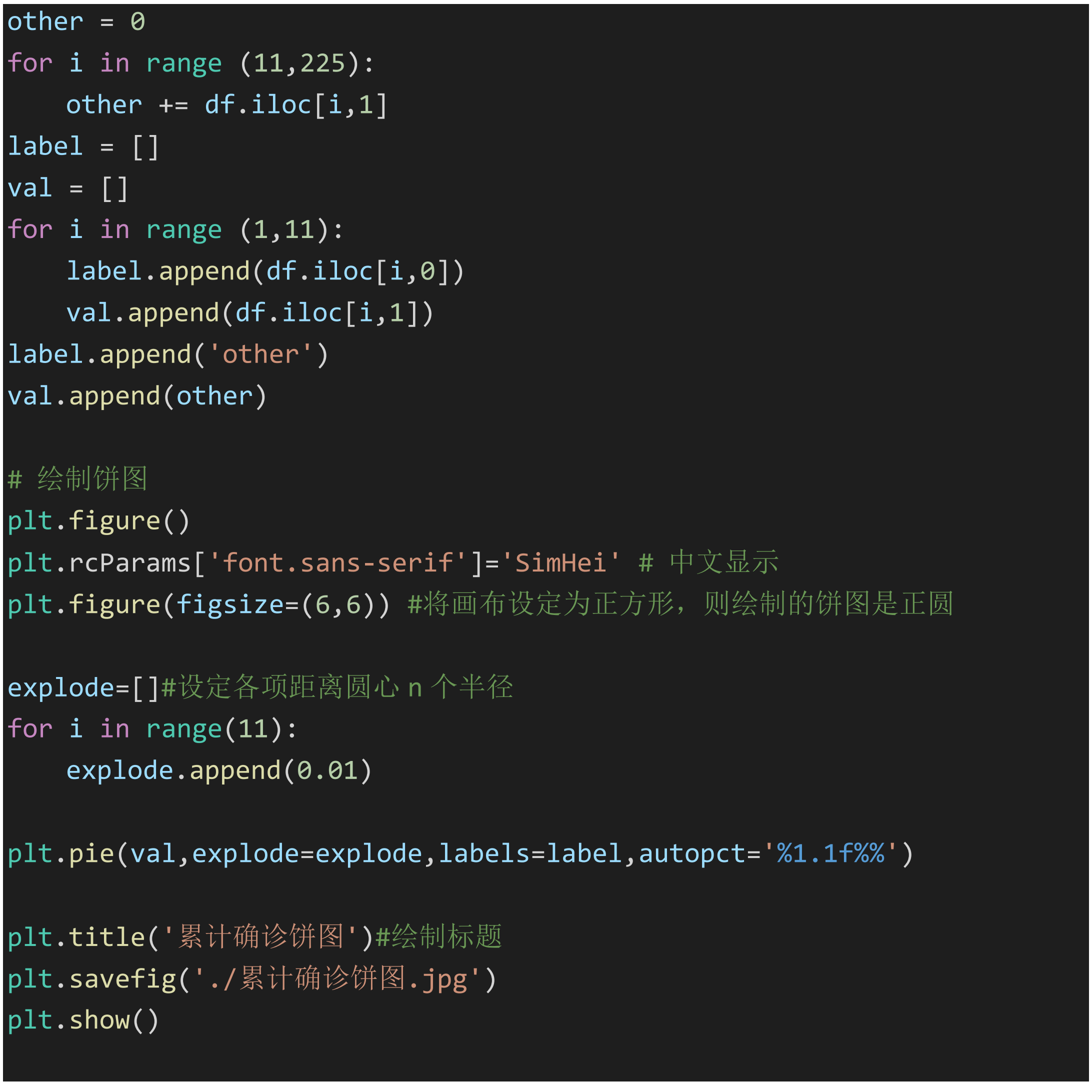


	A	B	C
1		国家	累计确诊
2	1	USA	60954028
3	2	India	35528004
4	3	Brazil	22499525
5	4	UK	14333794
6	5	France	11815121
7	6	Russia	10650849
8	7	Turkey	9916725
9	8	Germany	7500818
10	9	Italy	7281297
11	10	Spain	7164906



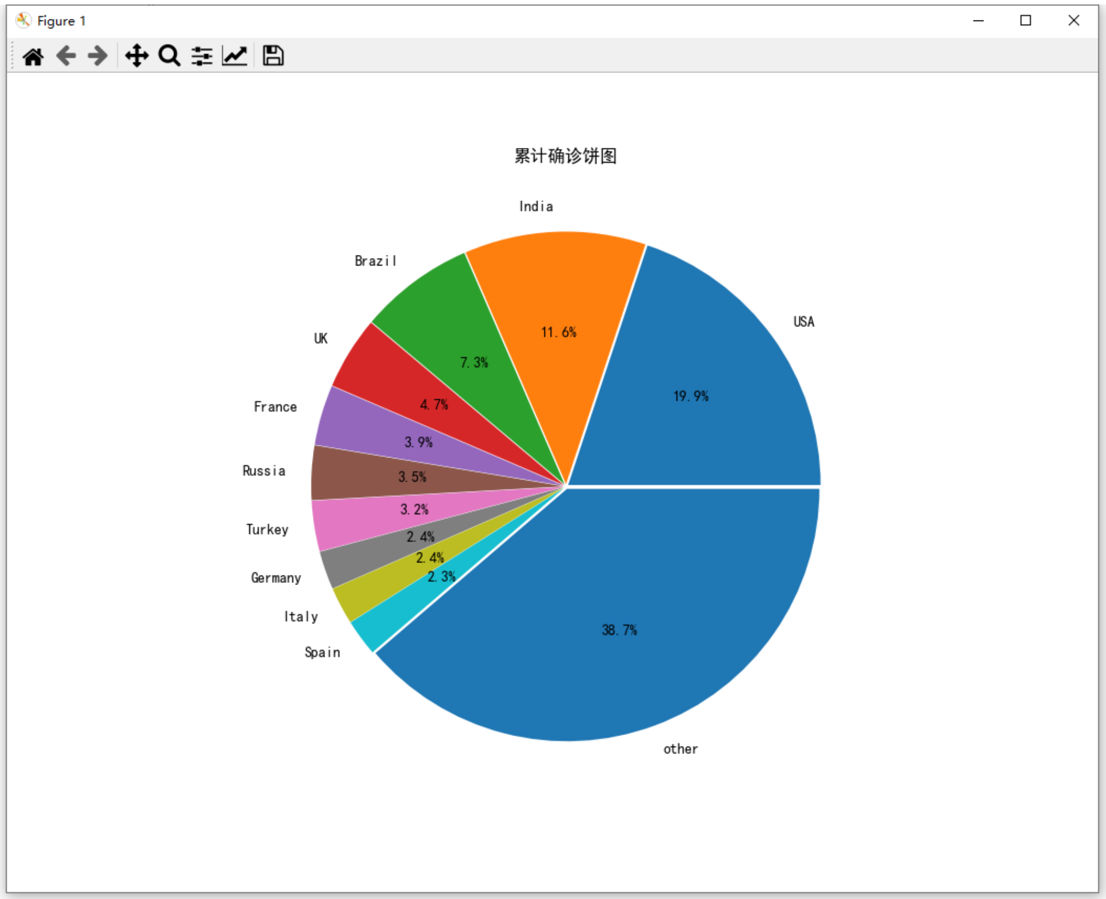
4）用饼图展示各个国家的累计确诊人数的比例

代码如下：



```
other = 0
for i in range (11,225):
    other += df.iloc[i,1]
label = []
val = []
for i in range (1,11):
    label.append(df.iloc[i,0])
    val.append(df.iloc[i,1])
label.append('other')
val.append(other)

# 绘制饼图
plt.figure()
plt.rcParams['font.sans-serif']='SimHei' # 中文显示
plt.figure(figsize=(6,6)) #将画布设定为正方形，则绘制的饼图是正圆

explode=[]#设定各项距离圆心 n 个半径
for i in range(11):
    explode.append(0.01)

plt.pie(val,explode=explode,labels=label,autopct='%1.1f%%')

plt.title('累计确诊饼图')#绘制标题
plt.savefig('./累计确诊饼图.jpg')
plt.show()
```

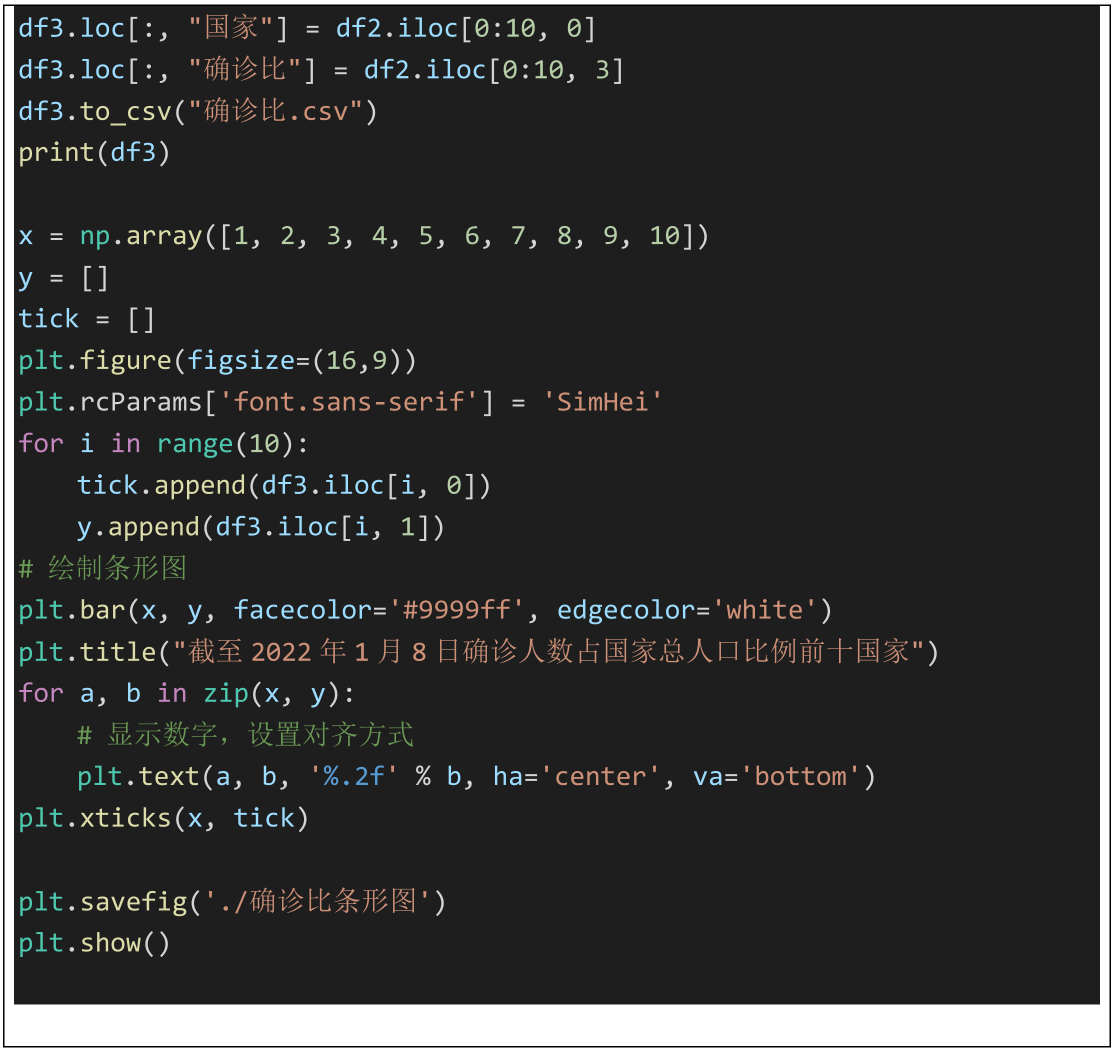
如图所示：



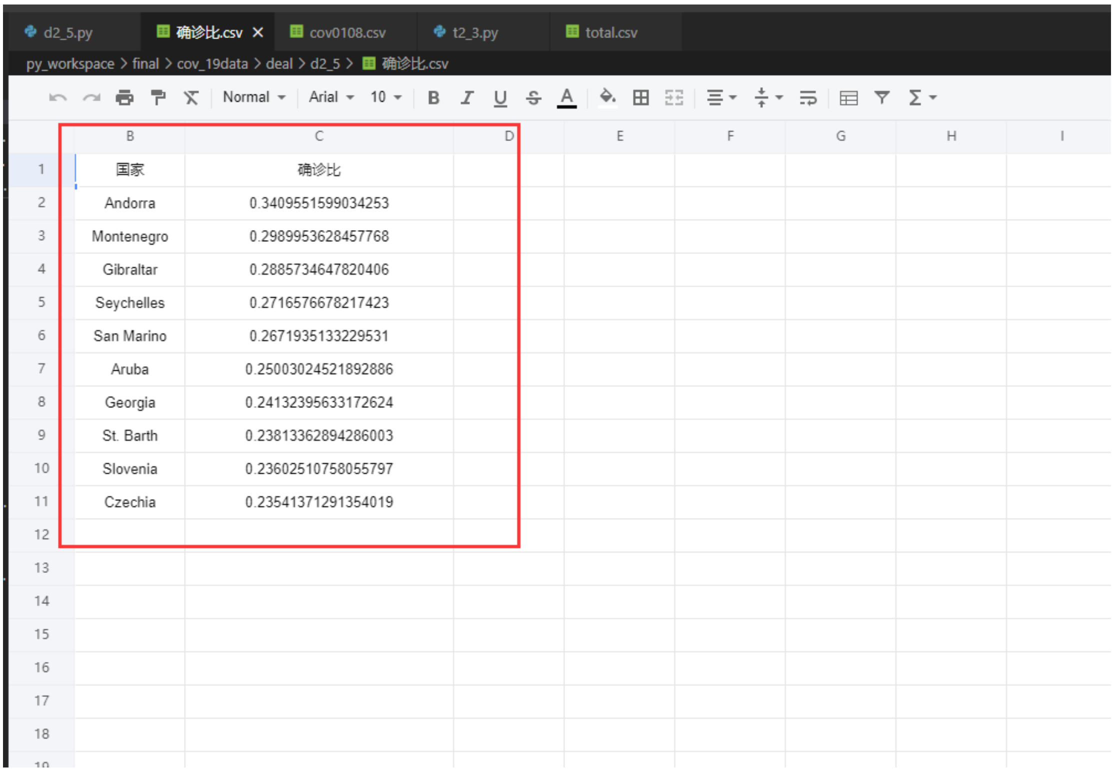
5）累计确诊人数占国家总人口比例最高的 10 个国家

代码如下：

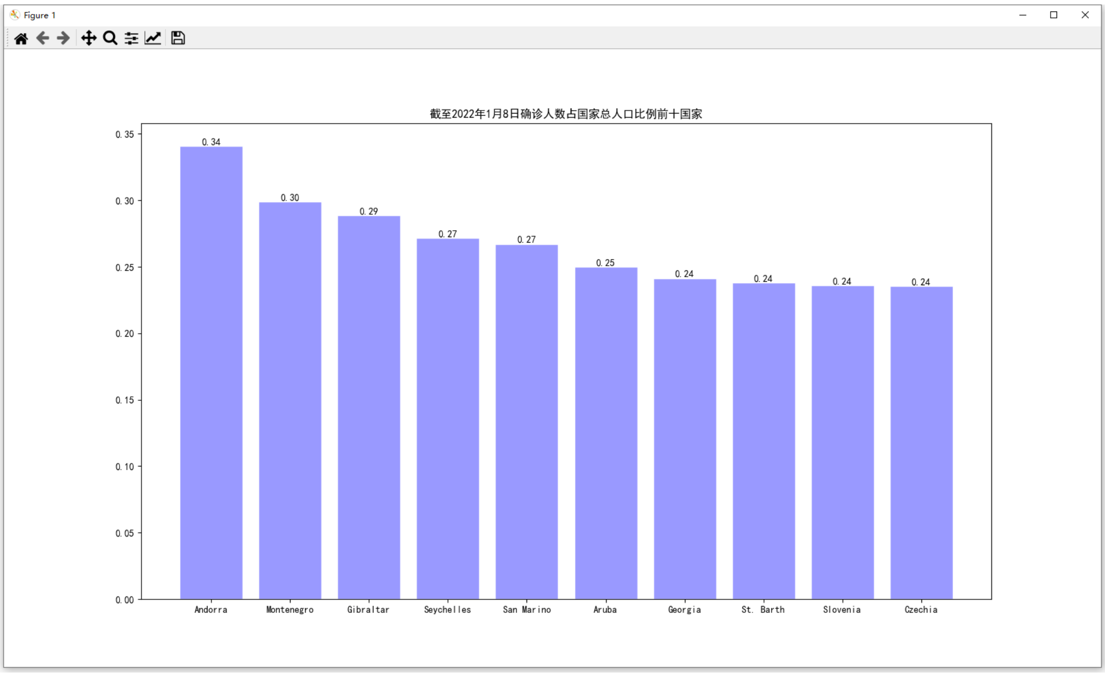
```
import numpy as np
import pandas as pd
import csv
import matplotlib.pylab as plt

fileName = "cov0108.csv"
df = pd.read_csv(fileName, encoding='utf-8', usecols=[1, 2, 6])

df2 = df.copy()
df2.insert(3, "确诊比", 0, True)
for i in range(1, 225):
    if df.loc[i, "人口总数"] > 0:
        df2.loc[i, "确诊比"] = df.loc[i, "累计确诊"] / df.loc[i, "人口总数"]

df2 = df2.sort_values(by="确诊比", ascending=False)

df3 = pd.DataFrame(columns=['国家', '确诊比'])
```



```
df3.loc[:, "国家"] = df2.iloc[0:10, 0]
df3.loc[:, "确诊比"] = df2.iloc[0:10, 3]
df3.to_csv("确诊比.csv")
print(df3)

x = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
y = []
tick = []
plt.figure(figsize=(16,9))
plt.rcParams['font.sans-serif'] = 'SimHei'
for i in range(10):
    tick.append(df3.iloc[i, 0])
    y.append(df3.iloc[i, 1])
# 绘制条形图
plt.bar(x, y, facecolor='#9999ff', edgecolor='white')
plt.title("截至2022年1月8日确诊人数占国家总人口比例前十国家")
for a, b in zip(x, y):
    # 显示数字，设置对齐方式
    plt.text(a, b, '%.2f' % b, ha='center', va='bottom')
plt.xticks(x, tick)

plt.savefig('./确诊比条形图')
plt.show()
```



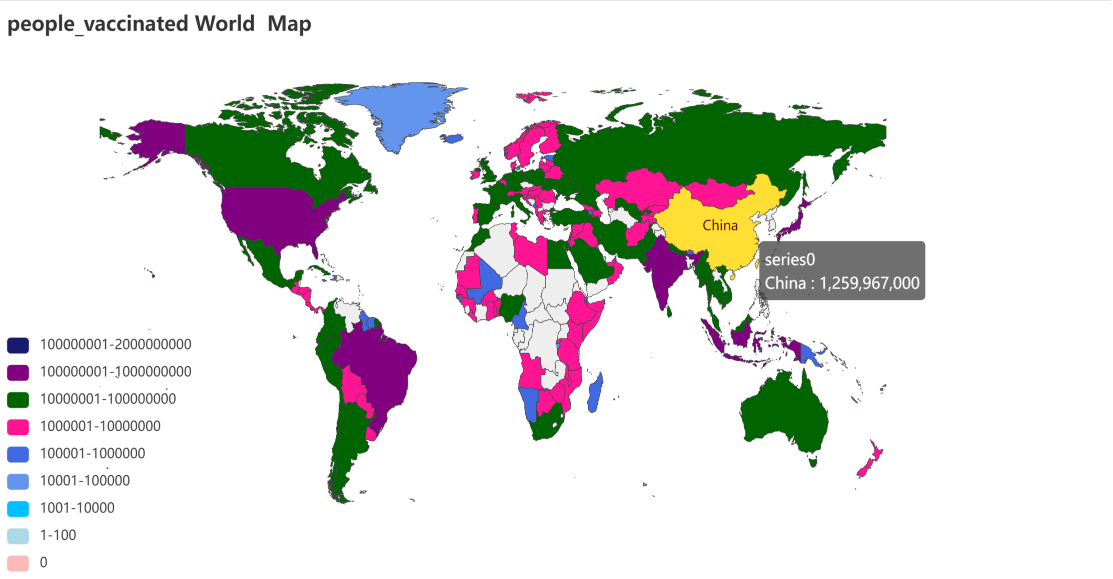
	B	C	D
1	国家	确诊比	
2	Andorra	0.3409551599034253	
3	Montenegro	0.2989953628457768	
4	Gibraltar	0.2885734647820406	
5	Seychelles	0.2716576678217423	
6	San Marino	0.2671935133229531	
7	Aruba	0.25003024521892886	
8	Georgia	0.24132395633172624	
9	St. Barth	0.23813362894286003	
10	Slovenia	0.23602510758055797	
11	Czechia	0.23541371291354019	



6）疫苗接种情况（至少接种了一针及以上），请用地图形式展示

代码如下：

```
from pyecharts.charts import Map  # 注意这里与老版本 pyecharts 调用的区别
from pyecharts import options as opts
import pandas as pd
import numpy as np
import random

# country = ['China', 'Canada', 'France', 'Japan', 'Russia', 'USA']
# data_world = [(i, random.randint(100, 200)) for i in country]

filename = 've.csv'

df = pd.read_csv(filename, encoding='utf-8', usecols=[0, 2])
data_world = []
for i in range(df.shape[0]):
    x = df.iloc[i, 0]
    y = df.iloc[i, 1]
    z = (x, y)
    data_world.append(z)
# print(data_world)

world = (
    Map().add(
        '',  # 此处没取名，所以空着
        data_world,  # 数据
        'world',
        is_map_symbol_show=False)  # 地图类型
    .set_global_opts(
        title_opts=opts.TitleOpts(title='people_vaccinated
World  Map'),
        visualmap_opts=opts.VisualMapOpts(max_=2000000000,
                                        is_piecewise=True,
                                        pieces=[
                                            {
                                                "max": 2000000000,
                                                "min": 1000000001,
                                                "label":
                                                "100000001-
2000000000",
                                                "color": "#191970"
                                            },
                                            {
                                                "max": 1000000000,
                                                "min": 100000001,
                                                "label":
                                                "100000001-
1000000000",
                                                "color": "#800080"
                                            },
                                            {
                                                "max": 100000000,
                                                "min": 10000001,
                                                "label":
                                                "10000001-100000000",
                                                "color": "#006400"
                                            },
                                            {
                                                "max": 10000000,
                                                "min": 1000001,
                                                "label": "1000001-
10000000",
                                                "color": "#FF1493"
                                            },
                                            {
                                                "max": 1000000,
                                                "min": 100001,
                                                "label": "100001-1000000",
                                                "color": "#4169E1"
                                            },
                                            {
                                                "max": 100000,
                                                "min": 10001,
                                                "label": "10001-100000",
                                                "color": "#6495ED"
                                            },
                                            {
                                                "max": 10000,
                                                "min": 1001,
                                                "label": "1001-10000",
                                                "color": "#00BFFF"
                                            },
                                            {
                                                "max": 1000,
                                                "min": 1,
                                                "label": "1-100",
                                                "color": "#ADD8E6"
                                            },
                                            {
                                                "max": 0,
                                                "min": 0,
                                                "label": "0",
                                                "color": "#fababa"
                                            },
                                        ])  # 定义图例为分段型，默认为连续的图例
    ).set_series_opts(label_opts=opts.LabelOpts(is_show=False))  # 国家名不显示
    .render(path='世界地图.html'))

# map = Map( init_opts=opts.InitOpts(width="1900px", height="900px", bg_color="#d0effa", page_title="全 xxxx_2"))
#     map.add("确 x 人数",[list(z) for z in zip(names_new, confirm)],is_map_symbol_show=False,
#              maptype="world",label_opts=opts.LabelOpts(is_show=False),itemstyle_opts=opts.ItemStyleOpts(color="rgb(98,121,146)"))#地图区域颜色
#     map.set_global_opts(title_opts = opts.TitleOpts(title='全 xxxx 诊人数'),legend_opts=opts.LegendOpts(is_show=False),
#                         visualmap_opts=opts.VisualMapOpts(max_=10000000, is_piecewise=True,
#                                         pieces=[
#                                           {"max": 10000000, "min": 100001, "label": ">1000", "color": "#8A0808"},
#                                           {"max": 100000, "min": 10001, "label": "500-1000", "color": "#B40404"},
#                                           {"max": 10000, "min": 1001, "label": "100-499", "color": "#DF0101"},
#                                           {"max": 1000, "min": 101, "label": "10-99", "color": "#F78181"},
#                                           {"max": 100, "min": 1, "label": "1-9", "color": "#F5A9A9"},
#                                           {"max": 0, "min": 0, "label": "0", "color": "#fababa"},
#                                           ])
#                         )
#     map.render('Global_new_crown_epidemic_map.html')
```

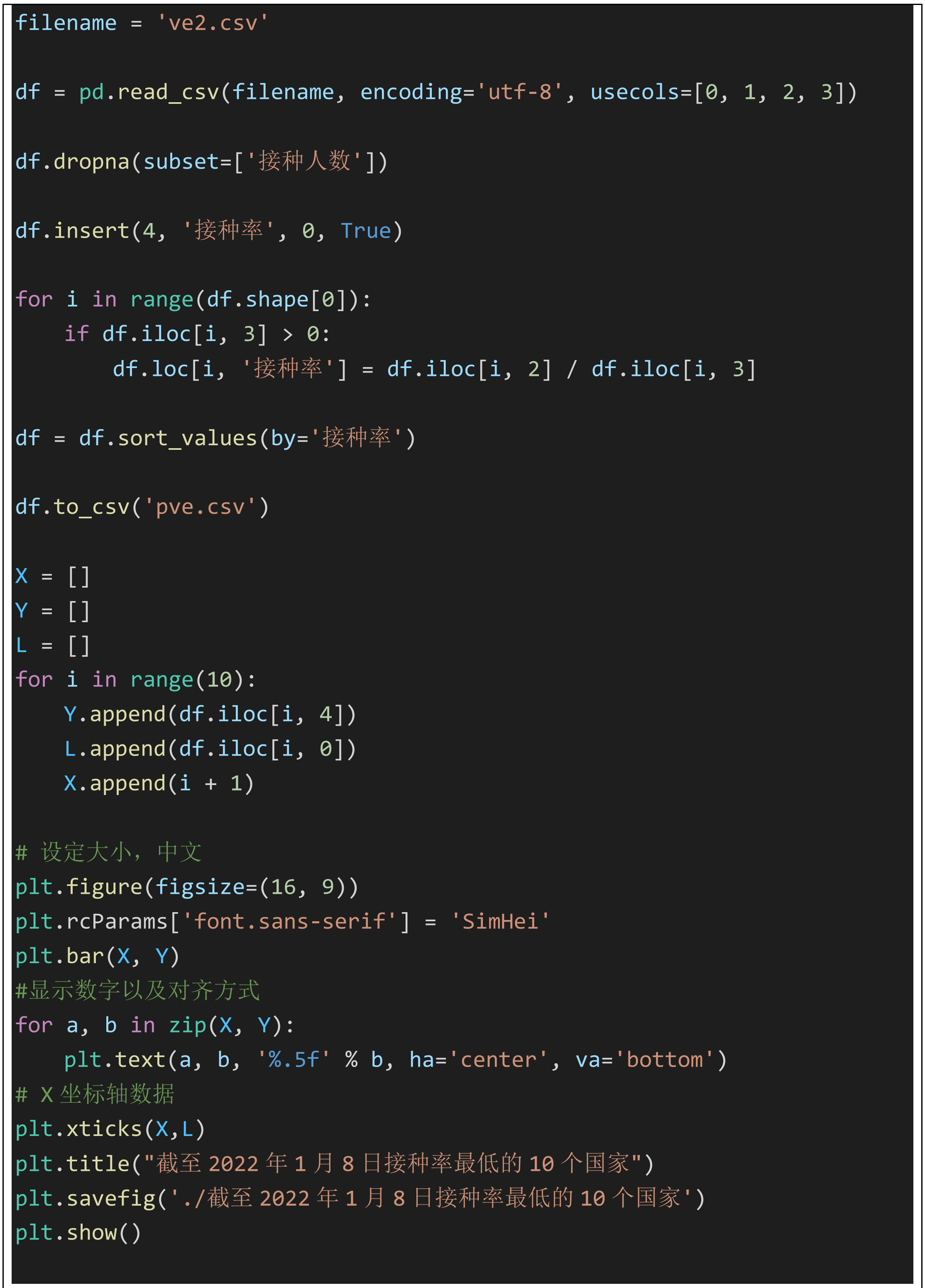


7）疫苗接种率累计疫苗接种人数/国家人数）最低的 10 个国家

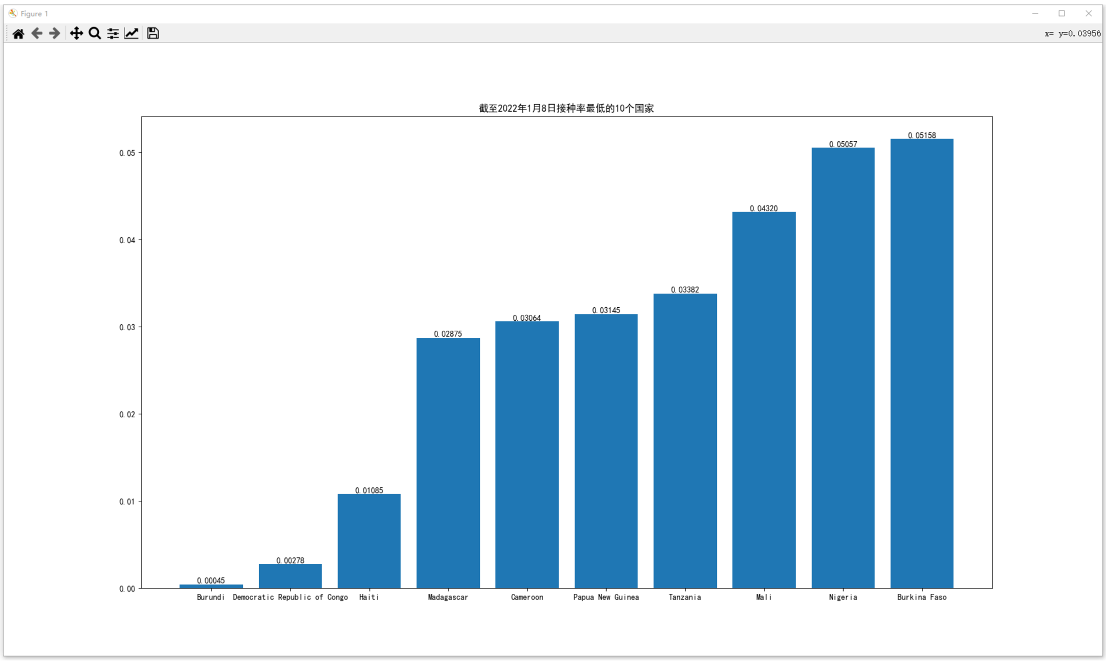
仅统计有疫苗接种数据的国家

代码如下：

```
from matplotlib.pyplot import colorbar
import numpy as np
import pandas as pd
import matplotlib.pylab as plt
```



```python
filename = 've2.csv'

df = pd.read_csv(filename, encoding='utf-8', usecols=[0, 1, 2, 3])

df.dropna(subset=['接种人数'])

df.insert(4, '接种率', 0, True)

for i in range(df.shape[0]):
    if df.iloc[i, 3] > 0:
        df.loc[i, '接种率'] = df.iloc[i, 2] / df.iloc[i, 3]

df = df.sort_values(by='接种率')

df.to_csv('pve.csv')

X = []
Y = []
L = []
for i in range(10):
    Y.append(df.iloc[i, 4])
    L.append(df.iloc[i, 0])
    X.append(i + 1)

# 设定大小，中文
plt.figure(figsize=(16, 9))
plt.rcParams['font.sans-serif'] = 'SimHei'
plt.bar(X, Y)
#显示数字以及对齐方式
for a, b in zip(X, Y):
    plt.text(a, b, '%.5f' % b, ha='center', va='bottom')
# X 坐标轴数据
plt.xticks(X,L)
plt.title("截至 2022 年 1 月 8 日接种率最低的 10 个国家")
plt.savefig('./截至 2022 年 1 月 8 日接种率最低的 10 个国家')
plt.show()
```



8）全球 GDP 前十名国家的累计确诊人数箱型图

查询资料得知 2020 年全球 GDP 前十名国家分别是：

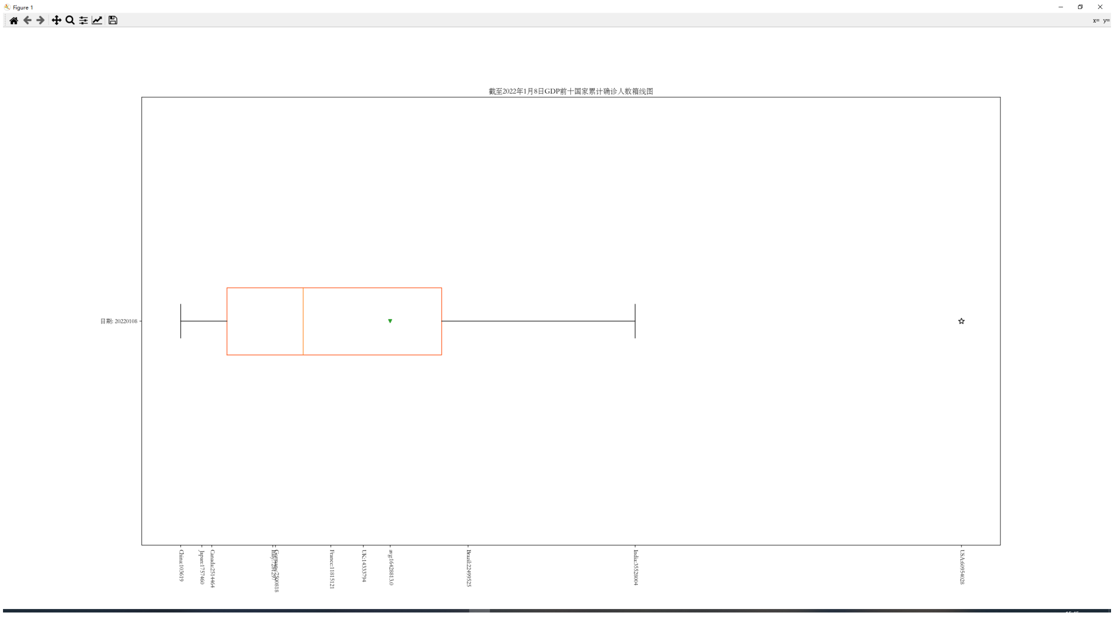
USA, China, Japan, Germany, India, UK, France, Italy, Brazil, Canada

代码如下：

```
from matplotlib.pyplot import colorbar
import numpy as np
import pandas as pd
import matplotlib.pylab as plt

fileName = "cov0108.csv"

df = pd.read_csv(fileName, encoding='utf-8', usecols=[1, 2])

# 2020 年 GDP 前十国家:
country = [
    'USA', 'China', 'Japan', 'Germany', 'India', 'UK', 'France', 
'Italy',
    'Brazil', 'Canada'
]

X = []
Y = []
L = []
j = 0
date = ['日期: 20220108']
for i in range(df.shape[0]):
    if df.iloc[i, 0] in country:
```

```python
        j += 1
        X.append(j)
        Y.append(df.iloc[i, 1])
        L.append(df.iloc[i, 0])

# 设置画布大小
plt.figure(figsize=(22, 10))
# 解决中文乱码
plt.rcParams['font.sans-serif'] = ['STSong']

plt.boxplot(
    Y,
    vert=False,
    # 显示均值
    showmeans=True,
    # 设置均值为绿色下三角符号
    meanprops={
        "marker": "v",
        'color': "green"
    },
    boxprops={'color': "orangered"},
    showfliers=True,
    flierprops={
        "marker": "*",
        "markersize": 10
    })
sum = 0
for i in range(len(Y)):
    sum += Y[i]
avg = sum / len(Y)

Y.append(avg)
L.append('avg')

for i in range(len(L)):
    L[i] = L[i] + ":" + str(Y[i])
plt.yticks([1], date)
# plt.xticks(Y,rotation=80)

plt.xticks(Y, L, rotation=-90)
plt.title("截至2022年1月8日GDP前十国家累计确诊人数箱线图")

plt.savefig("./GDP前十国家确诊人数箱线图")
plt.show()
```

这 10 个国家的累计确诊人数箱型图如下：
其中绿色倒三角形是平均数



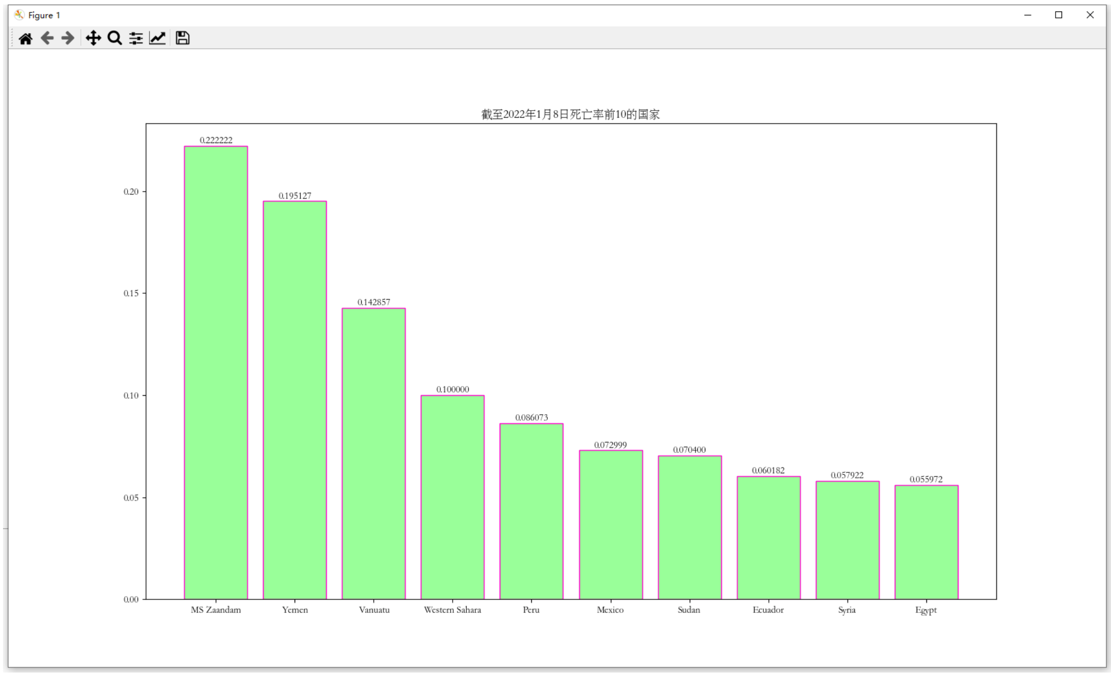
Italy:7281297
Germany:7500818

9）死亡率最高的 10 个国家

```
from matplotlib.pyplot import colorbar
import numpy as np
import pandas as pd
import matplotlib.pylab as plt

fileName = "cov0108.csv"

df = pd.read_csv(fileName, encoding='utf-8', usecols=[1, 2, 4, 6])

df.insert(4, "死亡率", 0, True)
for i in range(df.shape[0]):
    if df.loc[i, "累计确诊"] > 0:
        df.loc[i, "死亡率"] = df.loc[i, "累计死亡"] / df.loc[i, "累计确诊"]

df = df.sort_values(by = "死亡率",ascending=False)
```

```
df.to_csv("./死亡率.csv")
print(df)

X = []
Y = []
L =[]

# 解决中文乱码
plt.rcParams['font.sans-serif'] = ['STSong']

for i in range (10):
    L.append(df.iloc[i,0])
    Y.append(df.iloc[i,4])
    X.append(i+1)

plt.figure(figsize=(16,9))
plt.bar(X,Y,facecolor = '#99ff99',edgecolor = '#ff00cc')
for a,b in zip(X,Y):
    plt.text(a,b,'%.6f'% b ,ha = 'center',va = 'bottom')

plt.xticks(X,L)



plt.title("截至 2022 年 1 月 8 日死亡率前 10 的国家")
plt.savefig("./截至 2022 年 1 月 8 日死亡率前 10 的国家")
plt.show()
```

结果如图所示



三、列出全世界应对新冠疫情最好的 10 个国家，并说明你的理由

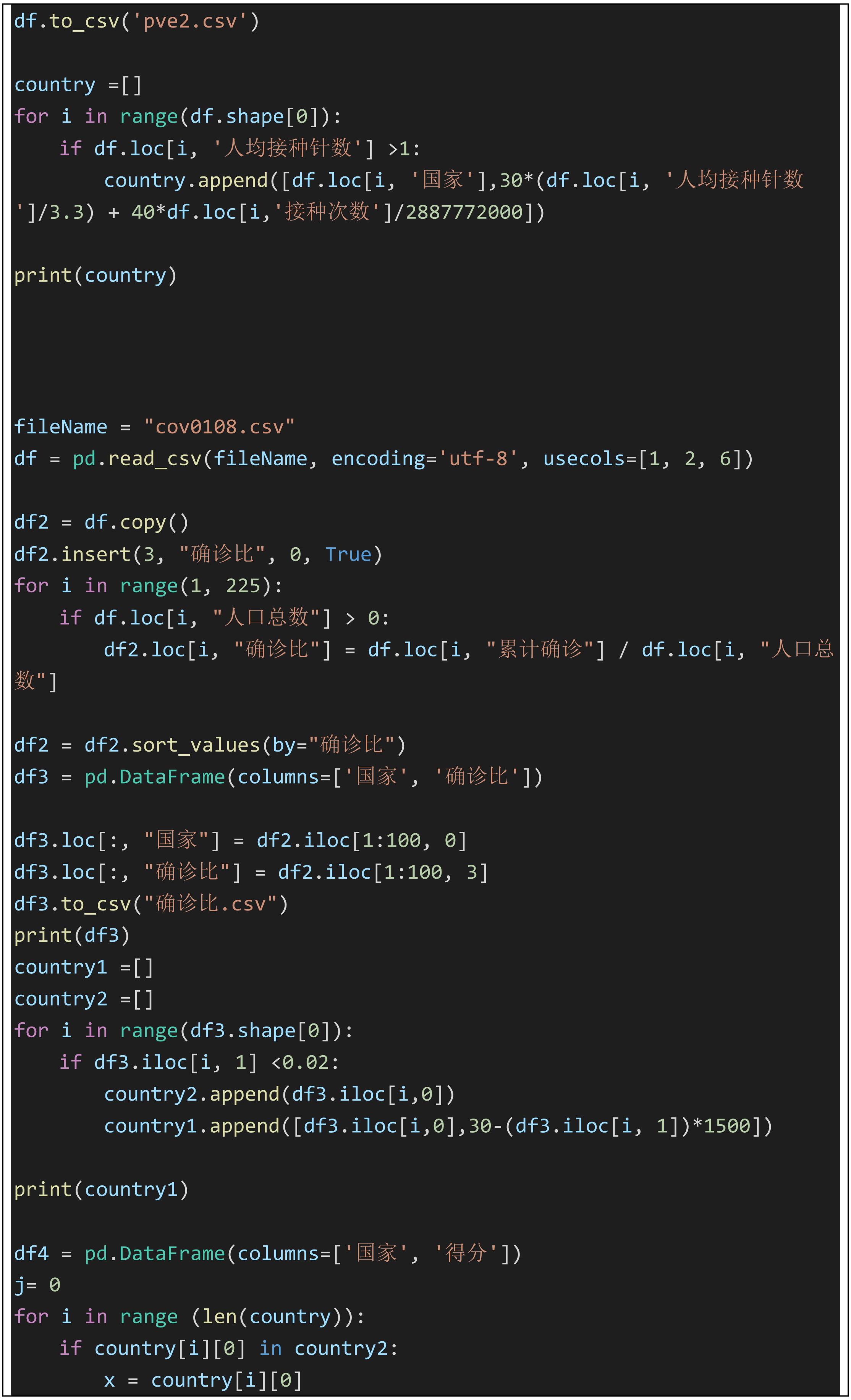
1. 人均疫苗接种数量需要超过 1 针

2. 确诊人数不超过国家人口的 0.1%

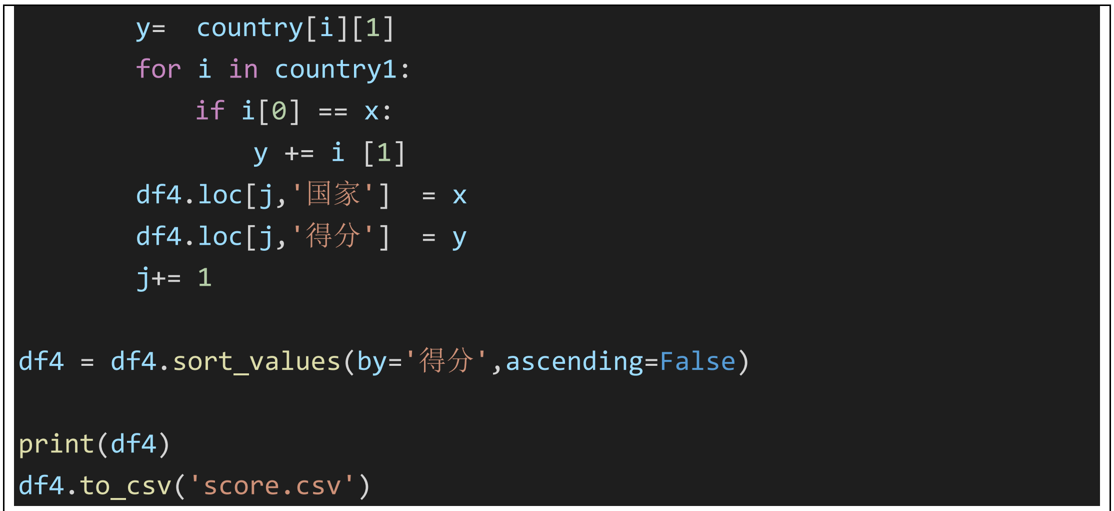
3. 按确诊人数占国家总人口数比例（30%），疫苗接种总数（40%），疫苗接种率（30%）综合排名，取前十名即为新冠疫情应对最好的国家

代码如下：

```
from matplotlib.pyplot import colorbar
import numpy as np
import pandas as pd
import matplotlib.pylab as plt

filename = 've2.csv'

df = pd.read_csv(filename, encoding='utf-8', usecols=[0, 1, 2, 3])

df.dropna(subset=['接种次数'])

df.insert(4, '人均接种针数', 0, True)

for i in range(df.shape[0]):
    if df.iloc[i, 3] > 0:
        df.loc[i, '人均接种针数'] = df.iloc[i, 1] / df.iloc[i, 3]

df = df.sort_values(by='人均接种针数',ascending=False)
```

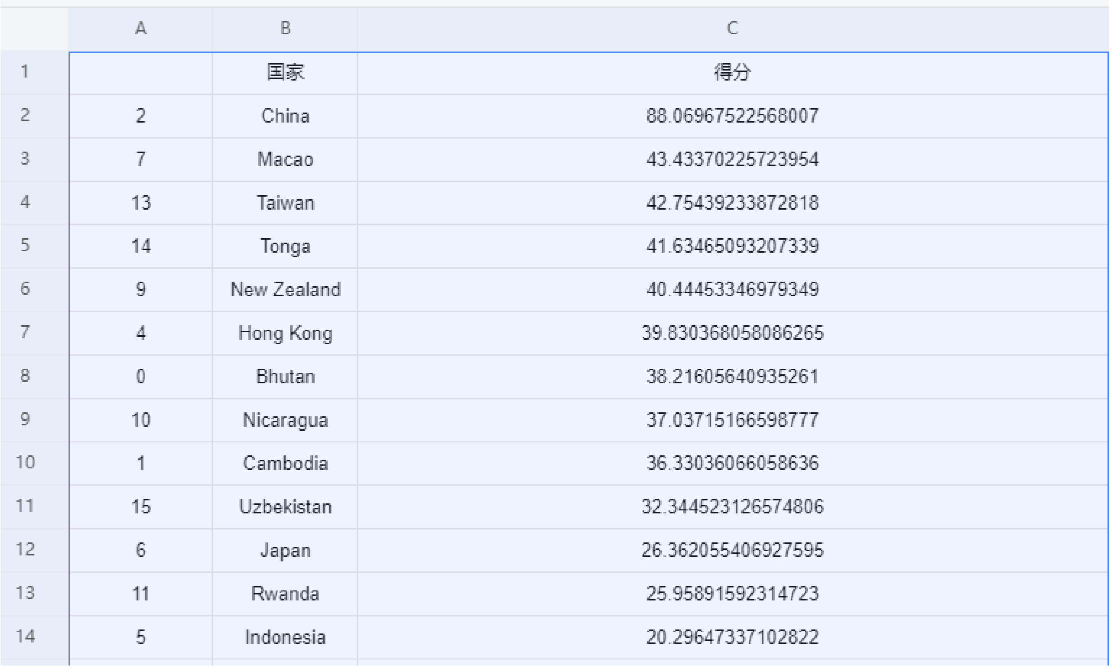


```python
df.to_csv('pve2.csv')

country =[]
for i in range(df.shape[0]):
    if df.loc[i, '人均接种针数'] >1:
        country.append([df.loc[i, '国家'],30*(df.loc[i, '人均接种针数']/3.3) + 40*df.loc[i,'接种次数']/2887772000])

print(country)




fileName = "cov0108.csv"
df = pd.read_csv(fileName, encoding='utf-8', usecols=[1, 2, 6])

df2 = df.copy()
df2.insert(3, "确诊比", 0, True)
for i in range(1, 225):
    if df.loc[i, "人口总数"] > 0:
        df2.loc[i, "确诊比"] = df.loc[i, "累计确诊"] / df.loc[i, "人口总数"]

df2 = df2.sort_values(by="确诊比")
df3 = pd.DataFrame(columns=['国家', '确诊比'])

df3.loc[:, "国家"] = df2.iloc[1:100, 0]
df3.loc[:, "确诊比"] = df2.iloc[1:100, 3]
df3.to_csv("确诊比.csv")
print(df3)
country1 =[]
country2 =[]
for i in range(df3.shape[0]):
    if df3.iloc[i, 1] <0.02:
        country2.append(df3.iloc[i,0])
        country1.append([df3.iloc[i,0],30-(df3.iloc[i, 1])*1500])

print(country1)

df4 = pd.DataFrame(columns=['国家', '得分'])
j= 0
for i in range (len(country)):
    if country[i][0] in country2:
        x = country[i][0]
```



```
        y=  country[i][1]
        for i in country1:
            if i[0] == x:
                y += i [1]
        df4.loc[j,'国家']  = x
        df4.loc[j,'得分']  = y
        j+= 1

df4 = df4.sort_values(by='得分',ascending=False)

print(df4)
df4.to_csv('score.csv')
```

结果如图所示:



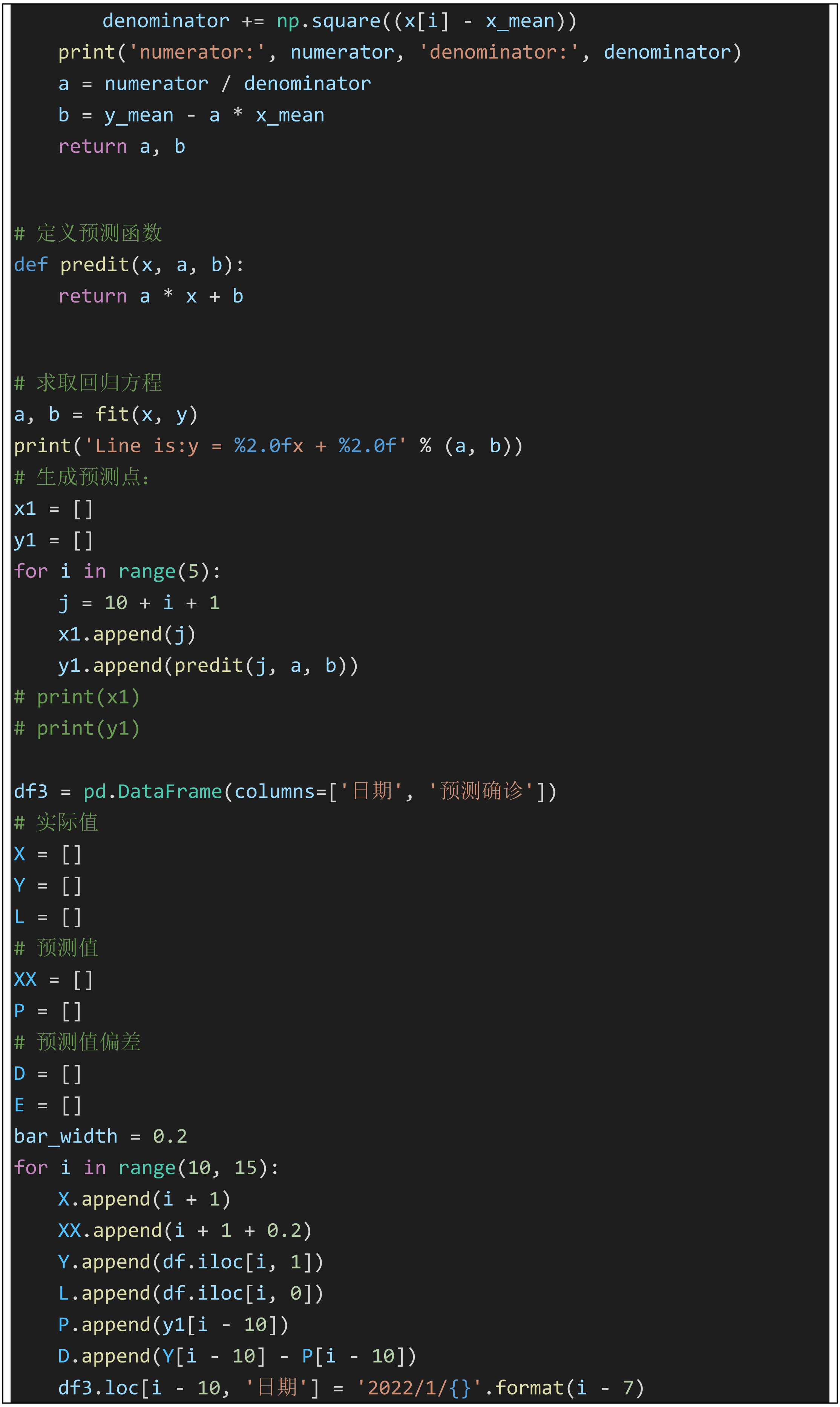
	A	B	C
1		国家	得分
2	2	China	88.06967522568007
3	7	Macao	43.43370225723954
4	13	Taiwan	42.75439233872818
5	14	Tonga	41.63465093207339
6	9	New Zealand	40.44453346979349
7	4	Hong Kong	39.830368058086265
8	0	Bhutan	38.21605640935261
9	10	Nicaragua	37.03715166598777
10	1	Cambodia	36.33036066058636
11	15	Uzbekistan	32.344523126574806
12	6	Japan	26.362055406927595
13	11	Rwanda	25.95891592314723
14	5	Indonesia	20.29647337102822

结果为：China，Tonga ， New Zealand ， Bhutan， Nicaragua， Cambodia， Uzbekistan，Japan， Rwanda， Indonesia 这十个国家

四、预测分析：利用前 10 天采集到的数据做后 5 天的预测，并与实际数据进行对比。说明你预测的方法，并分析与实际数据的差距和原因

代码如下：

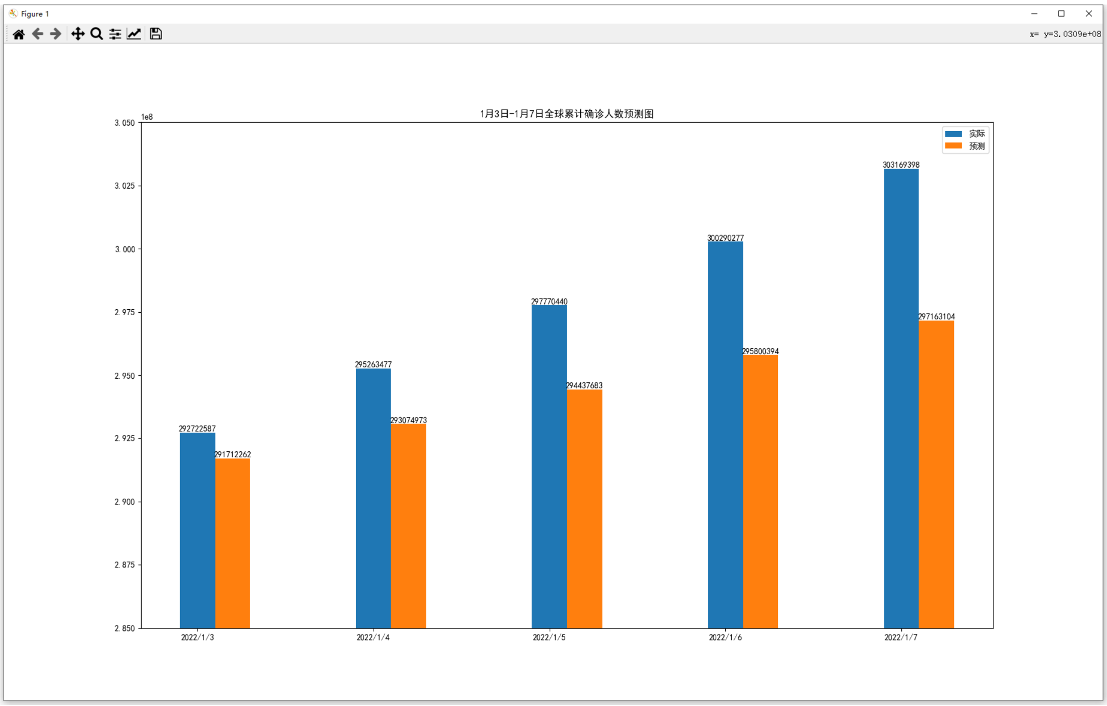
```
import pandas as pd
import numpy as np
import random
from matplotlib import pylab as plt

# 打开文件，读入数据
fileNmae = '15d_world.csv'
df = pd.read_csv(fileNmae, encoding='utf-8', usecols=[2, 3])

df2 = pd.DataFrame(columns=['序号', '日期', '累计确诊'])
# X = []
# Y = []
# L = []
# for i in range(10):
#     X.append(i + 1)
#     Y.append(df.iloc[i, 1])
#     L.append(df.iloc[i, 0])

for i in range(10):
    df2.loc[i, '序号'] = i + 1
    df2.loc[i, '日期'] = df.iloc[i, 0]
    df2.loc[i, '累计确诊'] = df.iloc[i, 1]
x = df2['序号']
y = df2['累计确诊']


# 由之前的条形图，数据近似是线性的
def fit(X, Y):
    if len(X) != len(Y):
        return
    numerator = 0.0  # 定义分子
    denominator = 0.0  #定义分母
    x_mean = np.mean(x)
    y_mean = np.mean(y)
    for i in range(len(x)):
        numerator += (x[i] - x_mean) * (y[i] - y_mean)
```



```
        denominator += np.square((x[i] - x_mean))
    print('numerator:', numerator, 'denominator:', denominator)
    a = numerator / denominator
    b = y_mean - a * x_mean
    return a, b


# 定义预测函数
def predit(x, a, b):
    return a * x + b


# 求取回归方程
a, b = fit(x, y)
print('Line is:y = %2.0fx + %2.0f' % (a, b))
# 生成预测点:
x1 = []
y1 = []
for i in range(5):
    j = 10 + i + 1
    x1.append(j)
    y1.append(predit(j, a, b))
# print(x1)
# print(y1)

df3 = pd.DataFrame(columns=['日期', '预测确诊'])
# 实际值
X = []
Y = []
L = []
# 预测值
XX = []
P = []
# 预测值偏差
D = []
E = []
bar_width = 0.2
for i in range(10, 15):
    X.append(i + 1)
    XX.append(i + 1 + 0.2)
    Y.append(df.iloc[i, 1])
    L.append(df.iloc[i, 0])
    P.append(y1[i - 10])
    D.append(Y[i - 10] - P[i - 10])
    df3.loc[i - 10, '日期'] = '2022/1/{}'.format(i - 7)
    df3.loc[i - 10, '预测确诊'] = y1[i - 10]
print(D)

plt.figure(figsize=(20, 12))
plt.rcParams['font.sans-serif'] = 'SimHei'
plt.bar(X, Y,width=bar_width, label='实际')
plt.bar(XX, P,width=bar_width, label='预测')
for a, b in zip(X, Y):
    plt.text(a, b, '%d' % b, ha='center', va='bottom')
for a, b in zip(XX, P):
    plt.text(a, b, '%d' % b, ha='center', va='bottom')
plt.legend(loc='upper right')
# X 坐标轴数据
plt.xticks(X, L)
plt.ylim((2.85e8, 3.05e8))
plt.title("1 月 3 日-1 月 7 日全球累计确诊人数预测图")
plt.savefig('./1 月 3 日-1 月 7 日全球累计确诊人数预测图')
plt.show()
```

假设数据是线性增长的，那么可计算的到线性回归方程：Line:y = 1362711x + 276722446

实际数据与预测数据对比如下：对比实际，每天偏差如下：1010324，2188503，3332756，4489882，6006293



可见还是有一些偏差，数据可能增长的稍快于线性

五、爬虫部分关键代码：

spider

```
import scrapy
from cov.items import CovItem


class CovidSpider(scrapy.Spider):
    name = 'covid'
    allowed_domains = ['www.worldometers.info']
    start_urls =
['https://www.worldometers.info/coronavirus/#main_table']

    def parse(self, response):
        j = 0
        i = -1
        item = CovItem()
        try:

            for each in response.xpath(
                    '//*[@id="main_table_countries_today"]/tbody[1]/*'
):
                j = j + 1
                # print(j)
                # item['country'] = each.xpath()
                content = each.xpath(
                    '//*[@id="main_table_countries_today"]/tbody[1]/tr
[{}]/td[2]//text()'
                    .format(j)).extract()

                # if j == 131:
                #     print(len(content))
                #     print(len(content[0]))
                #     print(content[0][0])
                #     for one in content:
                #         print(one)

                if len(content) > 0 and len(
                        content[0]) > 0 and content[0][0] != '\n':
                    ilist = []
                    i = i + 1
                    if i <= 225:
                        # print(i)
                        # print(content[0])
                        ilist.append(content[0])
                        for k in range(3, 16):
                            content = each.xpath(
                                '//*[@id="main_table_countries_today"]/
tbody[1]/tr[{}]/td[{}]//text()'
                                .format(j, k)).extract()
                            if len(content) > 0:
                                ilist.append(content[0])
                            else:
                                ilist.append('')
                        print(ilist)
                        try:
                            item['country'] = ilist[0]
                            item['total_cases'] = ilist[1]
                            item['new_cases'] = ilist[2]
                            item['total_death'] = ilist[3]
                            item['new_death'] = ilist[4]
                            item['total_recovered'] = ilist[5]
                            item['new_recovered'] = ilist[6]
                            item['active_cases'] = ilist[7]
                            item['serious'] = ilist[8]
                            item['tot_cases_per_m'] = ilist[9]
                            item['deaths_per_m'] = ilist[10]
                            item['total_tests'] = ilist[11]
                            item['tests_per_m'] = ilist[12]
                            item['population'] = ilist[13]
                            yield item
                            print("succeed yield")
                        except:
                            print("fail item")
        except ValueError:
            pass
```

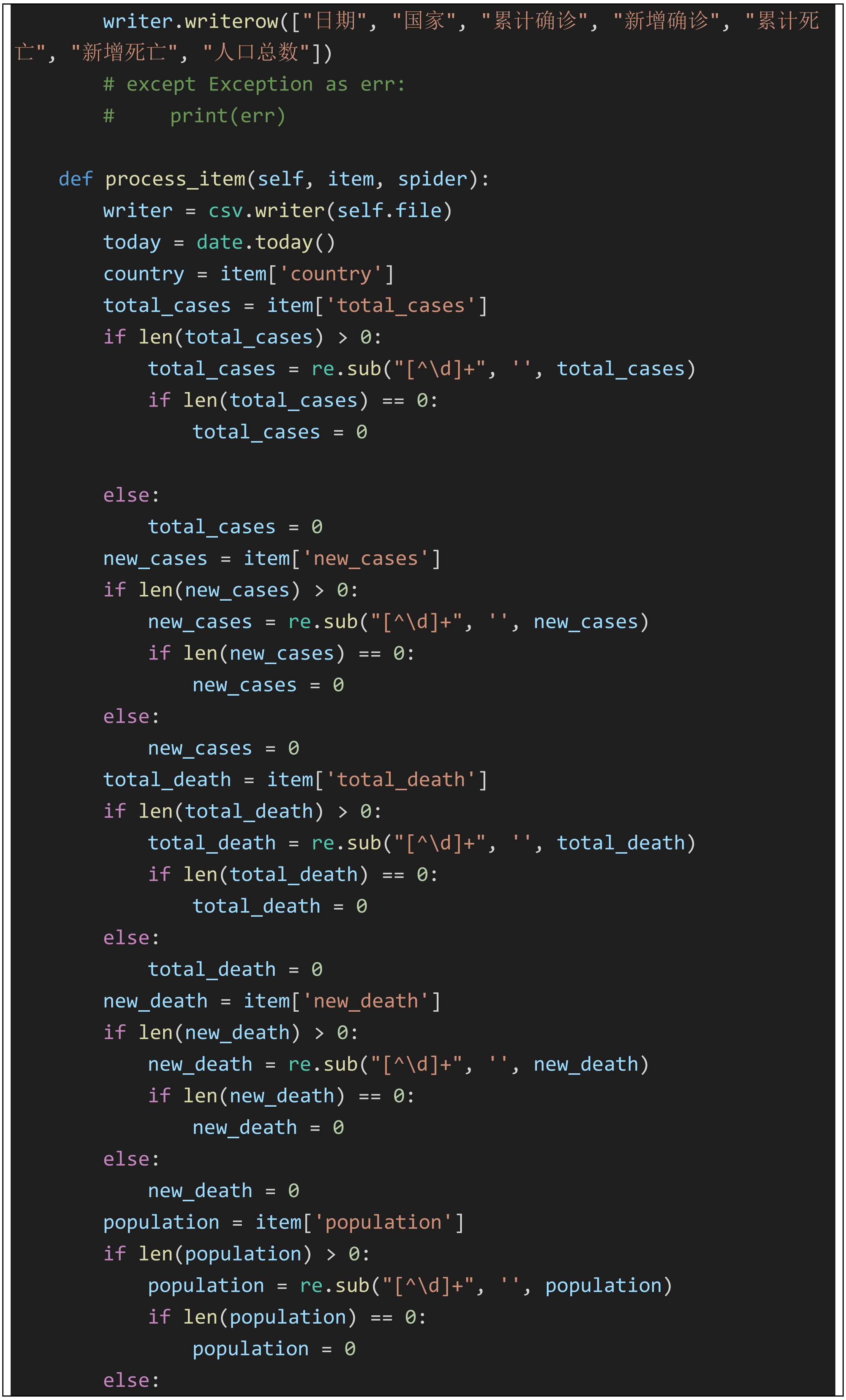
item

```
# Define here the models for your scraped items
#
# See documentation in:
# https://docs.scrapy.org/en/latest/topics/items.html

import scrapy
```

```


class CovItem(scrapy.Item):
    # define the fields for your item here like:
    # name = scrapy.Field()
    country = scrapy.Field()
    total_cases = scrapy.Field()
    new_cases = scrapy.Field()
    total_death = scrapy.Field()
    new_death = scrapy.Field()
    total_recovered = scrapy.Field()
    new_recovered = scrapy.Field()
    active_cases = scrapy.Field()
    serious = scrapy.Field()
    tot_cases_per_m = scrapy.Field()
    deaths_per_m = scrapy.Field()
    total_tests = scrapy.Field()
    tests_per_m = scrapy.Field()
    population = scrapy.Field()
```

pipeline

```
# Define your item pipelines here
#
# Don't forget to add your pipeline to the ITEM_PIPELINES setting
# See: https://docs.scrapy.org/en/latest/topics/item-pipeline.html

# useful for handling different item types with a single interface
import csv
from datetime import date, timedelta
import re


class CovPipeline:
    def open_spider(self, spider):
        # 开始爬虫，打开 csv 文件
        # today = date.today()
        # print(today)
        # fname = "cov0108.csv"
        self.file = open('cov0108.csv', 'w', newline='',
encoding='utf-8')
        writer = csv.writer(self.file)
```



```
        writer.writerow(["日期", "国家", "累计确诊", "新增确诊", "累计死亡", "新增死亡", "人口总数"])
        # except Exception as err:
        #     print(err)

    def process_item(self, item, spider):
        writer = csv.writer(self.file)
        today = date.today()
        country = item['country']
        total_cases = item['total_cases']
        if len(total_cases) > 0:
            total_cases = re.sub("[^\d]+", '', total_cases)
            if len(total_cases) == 0:
                total_cases = 0

        else:
            total_cases = 0
        new_cases = item['new_cases']
        if len(new_cases) > 0:
            new_cases = re.sub("[^\d]+", '', new_cases)
            if len(new_cases) == 0:
                new_cases = 0
        else:
            new_cases = 0
        total_death = item['total_death']
        if len(total_death) > 0:
            total_death = re.sub("[^\d]+", '', total_death)
            if len(total_death) == 0:
                total_death = 0
        else:
            total_death = 0
        new_death = item['new_death']
        if len(new_death) > 0:
            new_death = re.sub("[^\d]+", '', new_death)
            if len(new_death) == 0:
                new_death = 0
        else:
            new_death = 0
        population = item['population']
        if len(population) > 0:
            population = re.sub("[^\d]+", '', population)
            if len(population) == 0:
                population = 0
        else:
            population = 0
        if country == "World":
            population = ''
        # print(today)
        writer.writerow([
            "20220108", country, total_cases, new_cases,total_death,
            new_death, population
        ])

        # except:
        #     pass
        return item

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

settings

```
USER_AGENT = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71
Safari/537.36'
ROBOTSTXT_OBEY = False
ITEM_PIPELINES = {
   'cov.pipelines.CovPipeline': 300,
}
```