本次实验选择编程类工具Pyecharts。

在Pycharm中新建python项目，并引入pyecharts，openpyxl，datetime等必要包。

创建main.py，写入以下内容。

import time

import datetime

import openpyxl

from pyecharts.charts import Map, Timeline, Bar

from pyecharts import globals, options

globals.\_WarningControl.ShowWarning = False

filename = "CityData.xlsx"

ws = openpyxl.load\_workbook(filename)['CityData']

def subdays(date1: time, date2: time):

return (datetime.datetime(date1[0], date1[1], date1[2]) - datetime.datetime(date2[0], date2[1], date2[2])).days

def getMostLeftNear(a, b):

if (a > 0 and b > 0) or (a <= 0 and b <= 0):

return abs(a) < abs(b)

elif (a > 0 and b <= 0) or (a <= 0 and b > 0):

return b > 0

def confirmedAt(dateTime: time):

data, date = {}, {}

for i in range(2, ws.max\_row):

row = str(i + 1)

if date.\_\_contains\_\_(ws['B' + row].value):

dirdate\_april1 = subdays(time.strptime(

"2020-" + str(date[ws['B' + row].value]['month']) + "-" + str(date[ws['B' + row].value]['day']),

"%Y-%m-%d"), dateTime)

wsdate\_april1 = subdays(

time.strptime("2020-" + str(ws['H' + row].value.month) + "-" + str(ws['H' + row].value.day),

"%Y-%m-%d"), dateTime)

if date.\_\_contains\_\_(ws['B' + row].value) and getMostLeftNear(wsdate\_april1, dirdate\_april1):

date[ws['B' + row].value]['month'] = ws['H' + row].value.month

date[ws['B' + row].value]['day'] = ws['H' + row].value.day

elif not date.\_\_contains\_\_(ws['B' + row].value):

date[ws['B' + row].value] = {'month': ws['H' + row].value.month, 'day': ws['H' + row].value.day}

for i in range(2, ws.max\_row):

row = str(i + 1)

wsdate = ws['H' + row].value

if wsdate.month == date[ws['B' + row].value]['month'] and wsdate.day == date[ws['B' + row].value]['day']:

if data.\_\_contains\_\_(ws['B' + row].value):

data[ws['B' + row].value] += ws['D' + row].value

else:

data[ws['B' + row].value] = ws['D' + row].value

datalist = []

for k, v in data.items():

datalist.append([k.strip("省").strip("市").strip("自治区").strip("壮族").strip("回族").strip("维吾尔"), v])

return datalist

def drawConfirmedMapAt(dateTime: time):

map = Map()

map.set\_global\_opts(

title\_opts=options.TitleOpts(title=str(dateTime[1]) + '月' + str(dateTime[2]) + "日24时全国疫情确诊地图"),

visualmap\_opts=options.VisualMapOpts(max\_=3600, is\_piecewise=True,

pieces=[

{"max": 1999999, "min": 10000, "label": "10000人及以上",

"color": "#8A0808"},

{"max": 9999, "min": 1000, "label": "1000-9999人", "color": "#B40404"},

{"max": 999, "min": 500, "label": "500-999人", "color": "#DF0101"},

{"max": 499, "min": 100, "label": "100-499人", "color": "#F78181"},

{"max": 99, "min": 10, "label": "10-99人", "color": "#F5A9A9"},

{"max": 9, "min": 0, "label": "1-9人", "color": "#FFFFCC"},

], ) )

map.add("截止" + str(dateTime[1]) + '月' + str(dateTime[2]) + "日24时确诊数量", data\_pair=confirmedAt(dateTime),

maptype="china", is\_roam=True)

map.render(str(dateTime[1]) + '月' + str(dateTime[2]) + "日24时全国疫情确诊地图.html")

def getAllDate():

date = set()

for i in range(2, ws.max\_row):

curdate = ws['H' + str(i + 1)].value

date.add(time.strptime(str(curdate.year) + '-' + str(curdate.month) + '-' + str(curdate.day), "%Y-%m-%d"))

date = list(date)

date.sort()

return date

def drawAllConfirmedMap():

tl = Timeline()

date = getAllDate()

for curdate in date:

map = (

Map().add("截止" + str(curdate[1]) + '月' + str(curdate[2]) + "日24时确诊数量", data\_pair=confirmedAt(curdate),

maptype="china", is\_roam=True).set\_global\_opts(

title\_opts=options.TitleOpts(title=str(curdate[1]) + '月' + str(curdate[2]) + "日24时全国疫情确诊地图"),

visualmap\_opts=options.VisualMapOpts(max\_=3600, is\_piecewise=True,

pieces=[

{"max": 1999999, "min": 10000, "label": "10000人及以上",

"color": "#8A0808"},

{"max": 9999, "min": 1000, "label": "1000-9999人",

"color": "#B40404"},

{"max": 999, "min": 500, "label": "500-999人",

"color": "#DF0101"},

{"max": 499, "min": 100, "label": "100-499人",

"color": "#F78181"},

{"max": 99, "min": 10, "label": "10-99人", "color": "#F5A9A9"},

{"max": 9, "min": 0, "label": "1-9人", "color": "#FFFFCC"},

], ) # 最大数据范围，分段

)

)

tl.add(map, str(curdate[1]) + '月' + str(curdate[2]) + "日")

print(str(curdate[1]) + '月' + str(curdate[2]) + "日完成")

tl.render("全国疫情确诊地图.html")

def drawConfirmedBarAt(dateTime: time):

data = confirmedAt(dateTime)

title = str(dateTime[1]) + '月' + str(dateTime[2]) + "日24时全国累计确诊"

province, confirmed = [], []

for curdata in data:

province.append(curdata[0])

confirmed.append(curdata[1])

bar = Bar()

bar.add\_xaxis(province).add\_yaxis(

title, confirmed

).set\_global\_opts(

title\_opts=options.TitleOpts(title="Bar-基本示例", subtitle="我是副标题"),

xaxis\_opts=options.AxisOpts(axislabel\_opts=options.LabelOpts(rotate=-45)),

).render(

title + ".html"

)

print(title)

drawConfirmedBarAt(time.strptime("2020-04-01", "%Y-%m-%d"))

# drawConfirmedMapAt(time.strptime("2020-04-01", "%Y-%m-%d")) # 画出4月1号的疫情地图

drawAllConfirmedMap() # 画出所有时间的疫情地图

