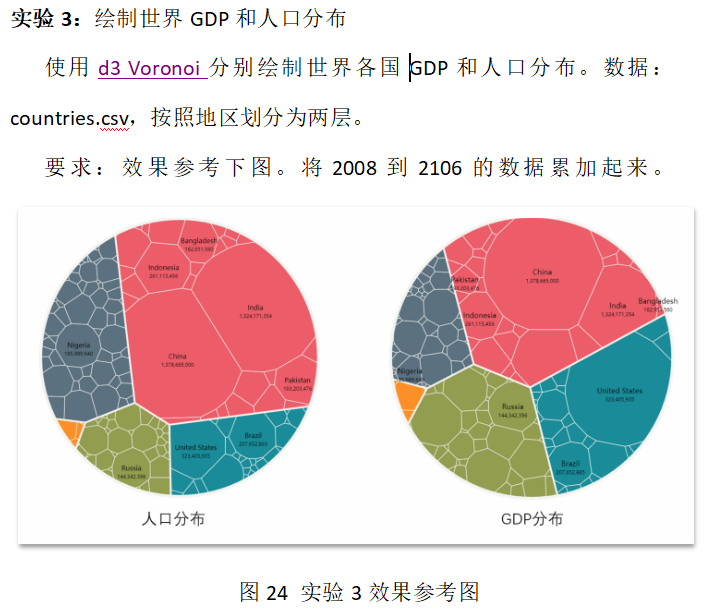
3022207128-杨宇鑫-实验报告4-3

1. 实验要求:

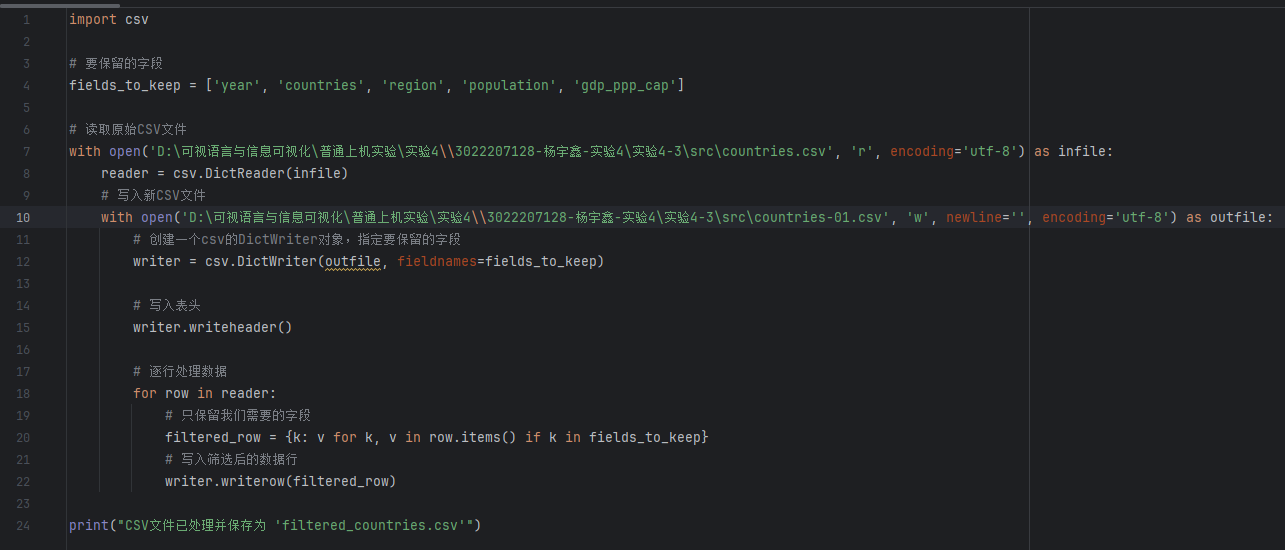


1. 实验过程:
   1. **处理数据**
      1. 编写python脚本处理数据

2.1.1.1 下面这么多的字段只保留year,countries,region,population,gdp\_ppp\_cap字段

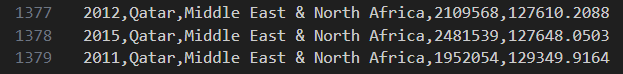
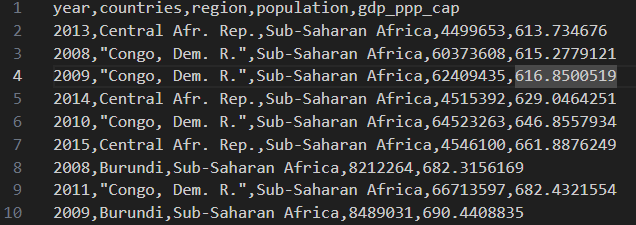
year,ISO\_code,countries,region,pf\_rol\_procedural,pf\_rol\_civil,pf\_rol\_criminal,pf\_rol,pf\_ss\_homicide,pf\_ss\_disappearances\_disap,pf\_ss\_disappearances\_violent,pf\_ss\_disappearances\_organized,pf\_ss\_disappearances\_fatalities,pf\_ss\_disappearances\_injuries,pf\_ss\_disappearances,pf\_ss\_women\_fgm,pf\_ss\_women\_missing,pf\_ss\_women\_inheritance\_widows,pf\_ss\_women\_inheritance\_daughters,pf\_ss\_women\_inheritance,pf\_ss\_women,pf\_ss,pf\_movement\_domestic,pf\_movement\_foreign,pf\_movement\_women,pf\_movement,pf\_religion\_estop\_establish,pf\_religion\_estop\_operate,pf\_religion\_estop,pf\_religion\_harassment,pf\_religion\_restrictions,pf\_religion,pf\_association\_association,pf\_association\_assembly,pf\_association\_political\_establish,pf\_association\_political\_operate,pf\_association\_political,pf\_association\_prof\_establish,pf\_association\_prof\_operate,pf\_association\_prof,pf\_association\_sport\_establish,pf\_association\_sport\_operate,pf\_association\_sport,pf\_association,pf\_expression\_killed,pf\_expression\_jailed,pf\_expression\_influence,pf\_expression\_control,pf\_expression\_cable,pf\_expression\_newspapers,pf\_expression\_internet,pf\_expression,pf\_identity\_legal,pf\_identity\_parental\_marriage,pf\_identity\_parental\_divorce,pf\_identity\_parental,pf\_identity\_sex\_male,pf\_identity\_sex\_female,pf\_identity\_sex,pf\_identity\_divorce,pf\_identity,pf\_score,pf\_rank,ef\_government\_consumption,ef\_government\_transfers,ef\_government\_enterprises,ef\_government\_tax\_income,ef\_government\_tax\_payroll,ef\_government\_tax,ef\_government,ef\_legal\_judicial,ef\_legal\_courts,ef\_legal\_protection,ef\_legal\_military,ef\_legal\_integrity,ef\_legal\_enforcement,ef\_legal\_restrictions,ef\_legal\_police,ef\_legal\_crime,ef\_legal\_gender,ef\_legal,ef\_money\_growth,ef\_money\_sd,ef\_money\_inflation,ef\_money\_currency,ef\_money,ef\_trade\_tariffs\_revenue,ef\_trade\_tariffs\_mean,ef\_trade\_tariffs\_sd,ef\_trade\_tariffs,ef\_trade\_regulatory\_nontariff,ef\_trade\_regulatory\_compliance,ef\_trade\_regulatory,ef\_trade\_black,ef\_trade\_movement\_foreign,ef\_trade\_movement\_capital,ef\_trade\_movement\_visit,ef\_trade\_movement,ef\_trade,ef\_regulation\_credit\_ownership,ef\_regulation\_credit\_private,ef\_regulation\_credit\_interest,ef\_regulation\_credit,ef\_regulation\_labor\_minwage,ef\_regulation\_labor\_firing,ef\_regulation\_labor\_bargain,ef\_regulation\_labor\_hours,ef\_regulation\_labor\_dismissal,ef\_regulation\_labor\_conscription,ef\_regulation\_labor,ef\_regulation\_business\_adm,ef\_regulation\_business\_bureaucracy,ef\_regulation\_business\_start,ef\_regulation\_business\_bribes,ef\_regulation\_business\_licensing,ef\_regulation\_business\_compliance,ef\_regulation\_business,ef\_regulation,ef\_score,ef\_rank,hf\_score,hf\_rank,hf\_quartile,population,gdp\_ppp\_cap,region\_simple,pop\_rank,gdp\_rank

python处理脚本:



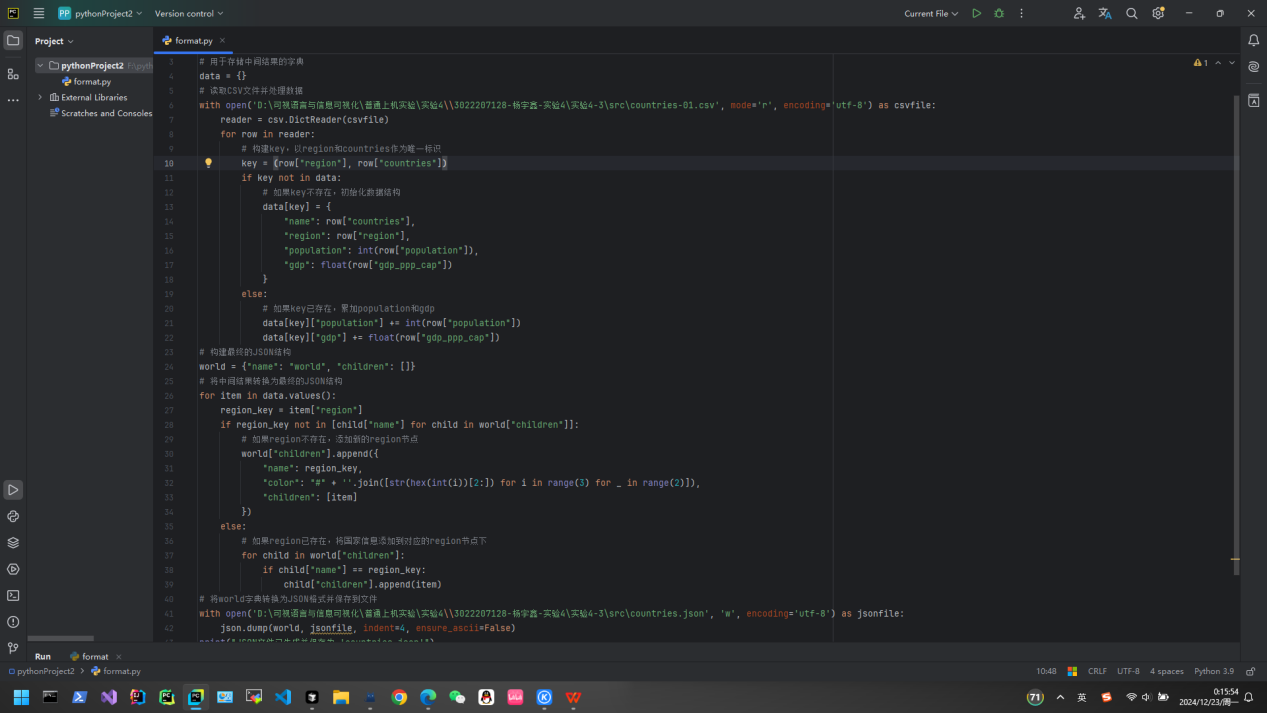
import csv  
# 要保留的字段  
fields\_to\_keep = ['year', 'countries', 'region', 'population', 'gdp\_ppp\_cap']  
# 读取原始CSV文件  
with open('D:\可视语言与信息可视化\普通上机实验\实验4\\3022207128-杨宇鑫-实验4\实验4-3\src\countries.csv', 'r', encoding='utf-8') as infile:  
 reader = csv.DictReader(infile)  
 # 写入新CSV文件  
 with open('D:\可视语言与信息可视化\普通上机实验\实验4\\3022207128-杨宇鑫-实验4\实验4-3\src\countries-01.csv', 'w', newline='', encoding='utf-8') as outfile:  
 # 创建一个csv的DictWriter对象，指定要保留的字段  
 writer = csv.DictWriter(outfile, fieldnames=fields\_to\_keep)  
 # 写入表头  
 writer.writeheader()  
 # 逐行处理数据  
 for row in reader:  
 # 只保留我们需要的字段  
 filtered\_row = {k: v for k, v in row.items() if k in fields\_to\_keep}  
 # 写入筛选后的数据行  
 writer.writerow(filtered\_row)  
print("CSV文件已处理并保存为 'filtered\_countries.csv'")

处理结果:



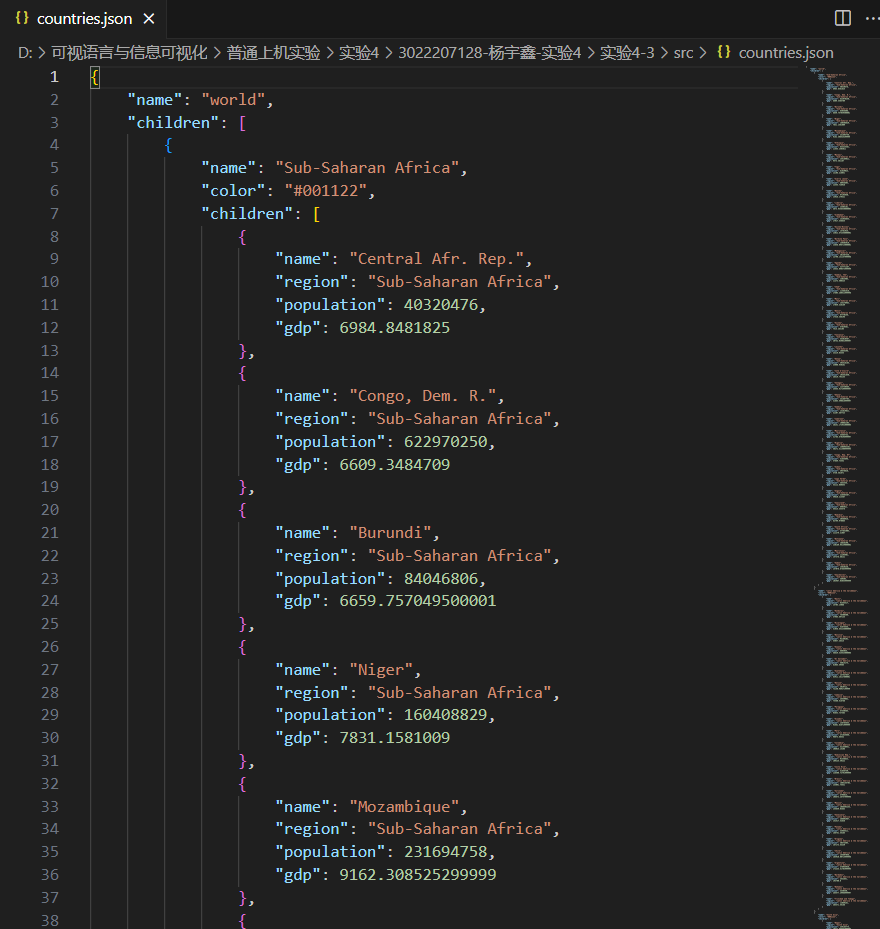
2.1.1.2 将csv格式文件用python脚本处理成json文件

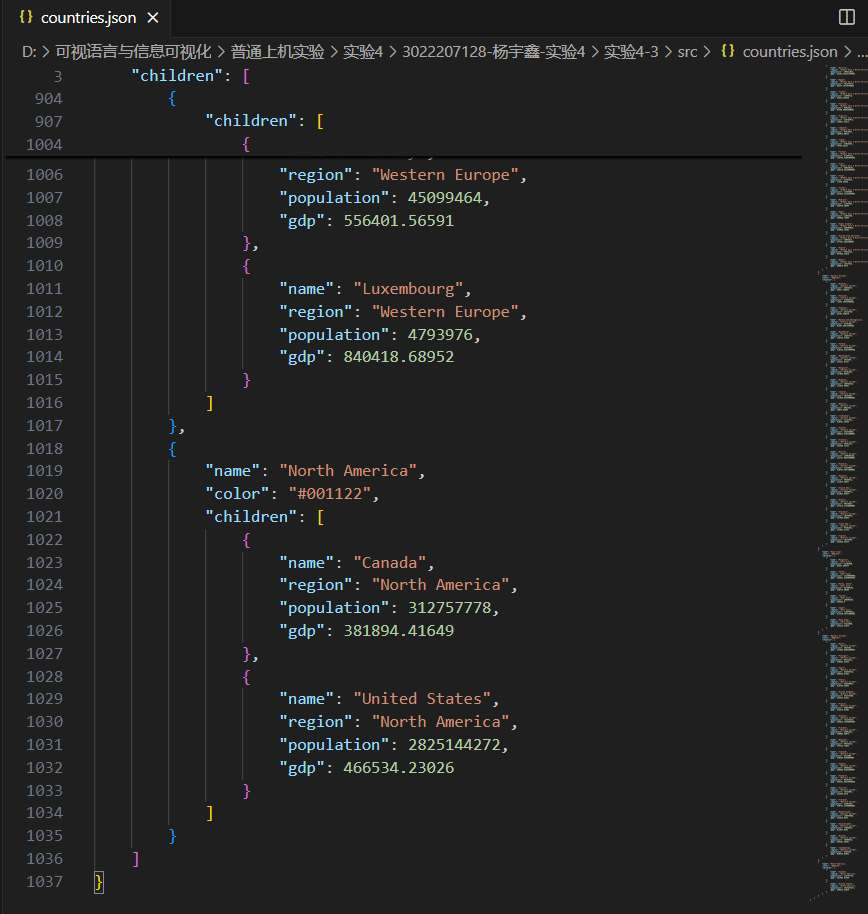
Python脚本:



import csv  
import json  
# 用于存储中间结果的字典  
data = {}  
# 读取CSV文件并处理数据  
with open('D:\可视语言与信息可视化\普通上机实验\实验4\\3022207128-杨宇鑫-实验4\实验4-3\src\countries-01.csv', mode='r', encoding='utf-8') as csvfile:  
 reader = csv.DictReader(csvfile)  
 for row in reader:  
 # 构建key，以region和countries作为唯一标识  
 key = (row["region"], row["countries"])  
 if key not in data:  
 # 如果key不存在，初始化数据结构  
 data[key] = {  
 "name": row["countries"],  
 "region": row["region"],  
 "population": int(row["population"]),  
 "gdp": float(row["gdp\_ppp\_cap"])  
 }  
 else:  
 # 如果key已存在，累加population和gdp  
 data[key]["population"] += int(row["population"])  
 data[key]["gdp"] += float(row["gdp\_ppp\_cap"])  
# 构建最终的JSON结构  
world = {"name": "world", "children": []}  
# 将中间结果转换为最终的JSON结构  
for item in data.values():  
 region\_key = item["region"]  
 if region\_key not in [child["name"] for child in world["children"]]:  
 # 如果region不存在，添加新的region节点  
 world["children"].append({  
 "name": region\_key,  
 "color": "#" + ''.join([str(hex(int(i))[2:]) for i in range(3) for \_ in range(2)]),  
 "children": [item]  
 })  
 else:  
 # 如果region已存在，将国家信息添加到对应的region节点下  
 for child in world["children"]:  
 if child["name"] == region\_key:  
 child["children"].append(item)  
# 将world字典转换为JSON格式并保存到文件  
with open('D:\可视语言与信息可视化\普通上机实验\实验4\\3022207128-杨宇鑫-实验4\实验4-3\src\countries.json', 'w', encoding='utf-8') as jsonfile:  
 json.dump(world, jsonfile, indent=4, ensure\_ascii=False)  
print("JSON文件已生成并保存为 'countries.json'")

处理结果:





2.2 编写代码

### 2.2.1. HTML 结构

在 index.html 和 index2.html 文件中，首先定义了基本的 HTML 结构，包括文档类型、头部信息和主体内容。每个文件中都有一个按钮，用于在两个页面之间切换。

<button onclick="location.href='index2.html'">人口分布</button>

### 2.2.2. 引入 D3.js 库

在 <head> 部分引入 D3.js 及其相关的插件，以便后续进行数据处理和可视化。

<script src="https://d3js.org/d3.v6.min.js" charset="utf-8"></script>  
<script src="https://rawcdn.githack.com/Kcnarf/d3-weighted-voronoi/v1.0.1/build/d3-weighted-voronoi.js"></script>  
<script src="https://rawcdn.githack.com/Kcnarf/d3-voronoi-map/v2.0.1/build/d3-voronoi-map.js"></script>  
<script src="https://rawcdn.githack.com/Kcnarf/d3-voronoi-treemap/v1.1.1/build/d3-voronoi-treemap.js"></script>

### 2.2.3. CSS 样式

在 <style> 标签中定义了一些基本的样式，包括 SVG 的大小、背景颜色、文本样式等。

body, html {  
 margin: 0;  
 overflow: hidden;  
}  
svg {  
 width: 100%;  
 height: 100%;  
 background-color: rgb(250, 250, 250);  
}

### 2.2.4. JavaScript 逻辑

在 <script> 标签中，主要实现了以下几个功能：

#### 2.2.4.1 常量定义

定义了一些常量，如 svgWidth 和 svgHeight，用于设置 SVG 的宽度和高度。

var svgWidth = window.innerWidth,  
 svgHeight = window.innerHeight;

#### 2.2.4.2 数据加载

使用 D3.js 的 d3.json 方法加载 countries.json 数据文件，并在数据加载完成后初始化数据和布局。

d3.json("countries.json").then(function (rootData) {  
 initData();  
 initLayout(rootData);  
});

#### 2.2.4.3 数据初始化

在 initData 函数中，计算出用于绘制的多边形，并设置字体缩放比例。

function initData(rootData) {  
 circlingPolygon = computeCirclingPolygon(treemapRadius);  
 fontScale.domain([3, 20]).range([8, 20]).clamp(true);  
}

#### 2.2.4.4 布局初始化

在 initLayout 函数中，创建 SVG 元素和绘图区域，并绘制世界的轮廓。

function initLayout(rootData) {  
 svg = d3.select("svg")  
 .attr("width", svgWidth)  
 .attr("height", svgHeight);  
 // 绘制世界轮廓  
 treemapContainer.append("path")  
 .classed("world", true)  
 .attr("d", "M" + circlingPolygon.join(",") + "Z");  
}

#### 2.2.4.5 绘制标题和图例

通过 drawTitle 和 drawLegends 函数绘制标题和图例，帮助用户理解图表内容。

function drawTitle() {  
 drawingArea.append("text")  
 .attr("id", "title")  
 .text("2008-2016世界各国人均GDP总值分布");  
}

#### 2.2.4.6 绘制树图

在 drawTreemap 函数中，使用 D3.js 的数据绑定和选择功能绘制树图，展示各国的 GDP 和人口数据。

function drawTreemap(hierarchy) {  
 var leaves = hierarchy.leaves();  
 var cells = treemapContainer.append("g")  
 .classed('cells', true)  
 .selectAll(".cell")  
 .data(leaves)  
 .enter()  
 .append("path")  
 .classed("cell", true)  
 .style("fill", function (d) {  
 return d.parent.data.color;  
 });  
}

## 实验结果

通过以上代码实现，用户可以在网页上看到一个动态的可视化地图，展示各国的人均 GDP 和人口分布。用户可以通过点击按钮在不同的页面之间切换，查看不同的数据可视化效果。

