1 Python程序设计#2作业

班级：307

学号：2021211138

姓名：陈朴炎

1.1 作业题目

数据文件（graph.txt）是一个全球温度年度异常历史数据。基于Sanic实现一个查询服务，服务包括：

- 按起始和结束年份查询历史数据，查询结果支持多种格式：JSON、XML、CSV（用逗号作为间隔 符）。
- 按温度高低进行排序，支持升序和降序两种排序方式。

1.2 作业内容

程序源代码：

```
from sanic import Sanic, response
import pandas as pd
import xml.etree.ElementTree as ET
app = Sanic(__name__)

data_file = "graph.txt"
df = pd.read_csv(data_file, sep="\s+", comment="#", names=["Year", "No_Smoothing",
"Lowess(5)"])

# 路由器根目录定义为 /
@app.route("/")
async def query_data(request):

    start_year = int(request.args.get("start_year", 1880))
    end_year = int(request.args.get("end_year", 2022))
    sort_by = request.args.get("sort_by", "year")  # 默认按年份排序
    order = request.args.get("order", "asc")    # 默认升序

    # 排序
    filtered_data = df[(df['Year'] >= start_year) & (df['Year'] <= end_year)]
    if sort_by == "temperature":
        filtered_data = filtered_data.sort_values(by='No_Smoothing', ascending=
(order == 'asc'))
    else:
        filtered_data = filtered_data.sort_values(by='Year' if order == "asc" else
'-Year')

    # 判断查询结果格式
    result_format = request.args.get("format", "json")

    if result_format == "json":
```

```
        result = filtered_data.to_dict(orient="records")
        return response.json(result)

    elif result_format == "xml":
        # 生成XML格式的文本响应
        xml_result = "<result>\n"
        for index, row in filtered_data.iterrows():
            year = int(row['Year'])
            no_smoothing = row['No_Smoothing']
            lowess = row['Lowess(5)']
            xml_result += f"\t<entry>\n\t\t<Year>{year}</Year>\n\t\t<No_Smoothing>
{no_smoothing:.2f}</No_Smoothing>\n\t\t<Lowess(5)>{lowess:.2f}
</Lowess(5)>\n\t</entry>\n"
        xml_result += "</result>"
        return response.text(xml_result, content_type="text/plain")

    elif result_format == "csv":
        text_result = ""
        for index, row in filtered_data.iterrows():
            year = int(row['Year'])
            text_result += f"{year},{row['No_Smoothing']:.2f},
{row['Lowess(5)']:.2f}\n"
        return response.text(text_result, content_type="text/plain")

    return response.text("Invalid format requested", status=400)

if __name__ == "__main__":
    app.run(port=8000)
```

1.3 代码说明

```
data_file = "graph.txt"
df = pd.read_csv(data_file, sep="\s+", comment="#", names=["Year", "No_Smoothing",
"Lowess(5)"])
```

这两行代码定义了首先定义了读入的文件名，下一行pd.read_csv函数将读取data_file的结果返回给df，其中读取的方式是csv文件格式，其中sep参数表示该文件的分隔符为一个或多个空格，comment表示这个文件的注释格式是以#开头的那一行，names代表表头，读取的每一行数据分别代表Year、No_Smoothing、Lowess(5)

```
@app.route("/")
```

表示把路由器的根目录定义为 / ，表示我将在我的网页后面直接写入查询语句。比如我的服务器网页是http://127.0.0.1:8000，那么我的查询语句query就会放在http://127.0.0.1:8000/?query里

```
async def query_data(request):
```

async是异步函数的定义，用于处理 HTTP 请求。async 关键字用于定义异步函数，这允许我这个查询函数在执行可能会耗费一些时间的操作时，让出 CPU 控制权，从而不会阻塞其他任务的执行。HTTP 请求会被传递给这个函数，在函数内部处理请求并生成响应。

```
    start_year = int(request.args.get("start_year", 1880))
    end_year = int(request.args.get("end_year", 2022))
    sort_by = request.args.get("sort_by", "year")  # 默认按年份排序
    order = request.args.get("order", "asc")    # 默认升序
```

这段代码用来获取请求中的起止年份和按哪个参数排序，怎么排序。开始年份如果没有，就默认为1880，结束年份如果没有，就设置成2022。排序的参考如果没有，就设置为year，默认为按年份排序，排序的升降序默认为升序。

```
    filtered_data = df[(df['Year'] >= start_year) & (df['Year'] <= end_year)]
    if sort_by == "temperature":
        filtered_data = filtered_data.sort_values(by='No_Smoothing', ascending=
(order == 'asc'))
    else:
        filtered_data = filtered_data.sort_values(by='Year' if order == "asc" else
'-Year')
```

这边先将数据按年份过滤出来，找出从start_year到end_year里的所有数据。之后看查询语句里是否有设置sort_by参数，如果设置了sort_by参数为temperature，那么就按照温度排序，否则就按照年份来排序。如果还设置了order参数，order参数如果不是asc的话就按照降序，如果没有设置order参数或者order参数为asc则默认为升序排序。

```
    result_format = request.args.get("format", "json")
```

这里我们将查询语句中的format参数提取出来，如果没有设置format的话，默认为json。

```
    if result_format == "json":
        result = filtered_data.to_dict(orient="records")
        return response.json(result)
```

这几行代码用来处理format为json的查询结果。将已经过滤排序好的数据通过json封装好，打印出来。

```
    elif result_format == "xml":
        # 生成XML格式的文本响应
        xml_result = "<result>\n"
        for index, row in filtered_data.iterrows():
            year = int(row['Year'])
            no_smoothing = row['No_Smoothing']
```

```
            lowess = row['Lowess(5)']
            xml_result += f"\t<entry>\n\t\t<Year>{year}</Year>\n\t\t<No_Smoothing>
{no_smoothing:.2f}</No_Smoothing>\n\t\t<Lowess(5)>{lowess:.2f}
</Lowess(5)>\n\t</entry>\n"
        xml_result += "</result>"
        return response.text(xml_result, content_type="text/plain")
```

这段代码用来处理format为xml的查询结果。首先查询结果包裹在 里面，每行结果都处在单独的子控件 ,entry里面又包括了、<No_Smoothing></No_Smoothing>、<Lowess(5)></Lowess(5)>，用来存放Year、No_Smoothing、Lowess(5)的信息。

```
    elif result_format == "csv":
        text_result = ""
        for index, row in filtered_data.iterrows():
            year = int(row['Year'])
            text_result += f"{year},{row['No_Smoothing']:.2f},
{row['Lowess(5)']:.2f}\n"
        return response.text(text_result, content_type="text/plain")
```

这段代码用来处理format为csv的格式，将每个信息都加入到文本中，用逗号分隔，最后返回。

```
return response.text("Invalid format requested", status=400)
```

要是不匹配的话就输出错误信息，状态信息设置为400

```
if __name__ == "__main__":
    app.run(port=8000)
```

主函数中将sanic服务器运行起来，放在本机的8000端口用来监听http请求。

1.4 运行结果

输入查询语句

```
http://localhost:8000/?
```

localhost:8000/?

[{"Year":1880,"No_Smoothing":-0.17,"Lowess(5)":-0.1},{"Year":1881,"No_Smoothing":-0.09,"Lowess(5)":-0.13},{"Year":1882,"No_Smoothing":-0.11,"Lowess(5)":-0.17},{"Year":1883,"No_Smoothing":-0.18,"Lowess(5)":-0.21},
{"Year":1884,"No_Smoothing":-0.29,"Lowess(5)":-0.24},{"Year":1885,"No_Smoothing":-0.34,"Lowess(5)":-0.27},{"Year":1886,"No_Smoothing":-0.32,"Lowess(5)":-0.28},{"Year":1887,"No_Smoothing":-0.37,"Lowess(5)":-0.28},
{"Year":1888,"No_Smoothing":-0.18,"Lowess(5)":-0.27},{"Year":1889,"No_Smoothing":-0.11,"Lowess(5)":-0.26},{"Year":1890,"No_Smoothing":-0.36,"Lowess(5)":-0.26},{"Year":1891,"No_Smoothing":-0.23,"Lowess(5)":-0.26},
{"Year":1892,"No_Smoothing":-0.28,"Lowess(5)":-0.27},{"Year":1893,"No_Smoothing":-0.32,"Lowess(5)":-0.27},{"Year":1894,"No_Smoothing":-0.32,"Lowess(5)":-0.25},{"Year":1895,"No_Smoothing":-0.24,"Lowess(5)":-0.23},
{"Year":1896,"No_Smoothing":-0.11,"Lowess(5)":-0.21},{"Year":1897,"No_Smoothing":-0.11,"Lowess(5)":-0.19},{"Year":1898,"No_Smoothing":-0.28,"Lowess(5)":-0.17},{"Year":1899,"No_Smoothing":-0.18,"Lowess(5)":-0.18},
{"Year":1900,"No_Smoothing":-0.08,"Lowess(5)":-0.2},{"Year":1901,"No_Smoothing":-0.16,"Lowess(5)":-0.24},{"Year":1902,"No_Smoothing":-0.28,"Lowess(5)":-0.26},{"Year":1903,"No_Smoothing":-0.37,"Lowess(5)":-0.28},
{"Year":1904,"No_Smoothing":-0.48,"Lowess(5)":-0.31},{"Year":1905,"No_Smoothing":-0.26,"Lowess(5)":-0.34},{"Year":1906,"No_Smoothing":-0.23,"Lowess(5)":-0.36},{"Year":1907,"No_Smoothing":-0.39,"Lowess(5)":-0.38},
{"Year":1908,"No_Smoothing":-0.43,"Lowess(5)":-0.39},{"Year":1909,"No_Smoothing":-0.49,"Lowess(5)":-0.41},{"Year":1910,"No_Smoothing":-0.44,"Lowess(5)":-0.41},{"Year":1911,"No_Smoothing":-0.44,"Lowess(5)":-0.39},
{"Year":1912,"No_Smoothing":-0.36,"Lowess(5)":-0.35},{"Year":1913,"No_Smoothing":-0.34,"Lowess(5)":-0.32},{"Year":1914,"No_Smoothing":-0.15,"Lowess(5)":-0.31},{"Year":1915,"No_Smoothing":-0.14,"Lowess(5)":-0.3},
{"Year":1916,"No_Smoothing":-0.36,"Lowess(5)":-0.3},{"Year":1917,"No_Smoothing":-0.46,"Lowess(5)":-0.3},{"Year":1918,"No_Smoothing":-0.3,"Lowess(5)":-0.3},{"Year":1919,"No_Smoothing":-0.28,"Lowess(5)":-0.29},
{"Year":1920,"No_Smoothing":-0.27,"Lowess(5)":-0.28},{"Year":1921,"No_Smoothing":-0.19,"Lowess(5)":-0.26},{"Year":1922,"No_Smoothing":-0.28,"Lowess(5)":-0.25},{"Year":1923,"No_Smoothing":-0.26,"Lowess(5)":-0.24},
{"Year":1924,"No_Smoothing":-0.27,"Lowess(5)":-0.23},{"Year":1925,"No_Smoothing":-0.22,"Lowess(5)":-0.22},{"Year":1926,"No_Smoothing":-0.11,"Lowess(5)":-0.22},{"Year":1927,"No_Smoothing":-0.22,"Lowess(5)":-0.21},
{"Year":1928,"No_Smoothing":-0.2,"Lowess(5)":-0.2},{"Year":1929,"No_Smoothing":-0.36,"Lowess(5)":-0.19},{"Year":1930,"No_Smoothing":-0.16,"Lowess(5)":-0.19},{"Year":1931,"No_Smoothing":-0.09,"Lowess(5)":-0.19},
{"Year":1932,"No_Smoothing":-0.16,"Lowess(5)":-0.18},{"Year":1933,"No_Smoothing":-0.28,"Lowess(5)":-0.17},{"Year":1934,"No_Smoothing":-0.12,"Lowess(5)":-0.15},{"Year":1935,"No_Smoothing":-0.2,"Lowess(5)":-0.14},
{"Year":1936,"No_Smoothing":-0.14,"Lowess(5)":-0.1},{"Year":1937,"No_Smoothing":-0.03,"Lowess(5)":-0.06},{"Year":1938,"No_Smoothing":0.0,"Lowess(5)":-0.01},{"Year":1939,"No_Smoothing":-0.02,"Lowess(5)":0.03},
{"Year":1940,"No_Smoothing":0.13,"Lowess(5)":0.07},{"Year":1941,"No_Smoothing":0.19,"Lowess(5)":0.09},{"Year":1942,"No_Smoothing":0.07,"Lowess(5)":0.11},{"Year":1943,"No_Smoothing":0.09,"Lowess(5)":0.1},
{"Year":1944,"No_Smoothing":0.21,"Lowess(5)":0.08},{"Year":1945,"No_Smoothing":0.09,"Lowess(5)":0.04},{"Year":1946,"No_Smoothing":-0.07,"Lowess(5)":0.01},{"Year":1947,"No_Smoothing":-0.02,"Lowess(5)":-0.03},
{"Year":1948,"No_Smoothing":-0.1,"Lowess(5)":-0.07},{"Year":1949,"No_Smoothing":-0.11,"Lowess(5)":-0.08},{"Year":1950,"No_Smoothing":-0.17,"Lowess(5)":-0.07},{"Year":1951,"No_Smoothing":-0.07,"Lowess(5)":-0.07},
{"Year":1952,"No_Smoothing":0.01,"Lowess(5)":-0.07},{"Year":1953,"No_Smoothing":0.08,"Lowess(5)":-0.07},{"Year":1954,"No_Smoothing":-0.13,"Lowess(5)":-0.06},{"Year":1955,"No_Smoothing":-0.14,"Lowess(5)":-0.05},
{"Year":1956,"No_Smoothing":-0.19,"Lowess(5)":-0.05},{"Year":1957,"No_Smoothing":0.05,"Lowess(5)":-0.04},{"Year":1958,"No_Smoothing":0.06,"Lowess(5)":-0.01},{"Year":1959,"No_Smoothing":0.03,"Lowess(5)":0.01},
{"Year":1960,"No_Smoothing":-0.02,"Lowess(5)":0.03},{"Year":1961,"No_Smoothing":0.06,"Lowess(5)":0.01},{"Year":1962,"No_Smoothing":0.03,"Lowess(5)":-0.01},{"Year":1963,"No_Smoothing":0.05,"Lowess(5)":-0.03},
{"Year":1964,"No_Smoothing":-0.2,"Lowess(5)":-0.04},{"Year":1965,"No_Smoothing":-0.11,"Lowess(5)":-0.05},{"Year":1966,"No_Smoothing":-0.06,"Lowess(5)":-0.06},{"Year":1967,"No_Smoothing":-0.02,"Lowess(5)":-0.05},
{"Year":1968,"No_Smoothing":-0.08,"Lowess(5)":-0.03},{"Year":1969,"No_Smoothing":0.05,"Lowess(5)":-0.02},{"Year":1970,"No_Smoothing":0.03,"Lowess(5)":-0.0},{"Year":1971,"No_Smoothing":-0.08,"Lowess(5)":0.0},
{"Year":1972,"No_Smoothing":0.01,"Lowess(5)":0.0},{"Year":1973,"No_Smoothing":0.16,"Lowess(5)":-0.0},{"Year":1974,"No_Smoothing":-0.07,"Lowess(5)":0.01},{"Year":1975,"No_Smoothing":-0.01,"Lowess(5)":0.02},
{"Year":1976,"No_Smoothing":-0.1,"Lowess(5)":0.04},{"Year":1977,"No_Smoothing":0.18,"Lowess(5)":0.07},{"Year":1978,"No_Smoothing":0.07,"Lowess(5)":0.12},{"Year":1979,"No_Smoothing":0.16,"Lowess(5)":0.16},
{"Year":1980,"No_Smoothing":0.26,"Lowess(5)":0.2},{"Year":1981,"No_Smoothing":0.32,"Lowess(5)":0.21},{"Year":1982,"No_Smoothing":0.14,"Lowess(5)":0.21},{"Year":1983,"No_Smoothing":0.31,"Lowess(5)":0.21},
{"Year":1984,"No_Smoothing":0.15,"Lowess(5)":0.21},{"Year":1985,"No_Smoothing":0.12,"Lowess(5)":0.22},{"Year":1986,"No_Smoothing":0.18,"Lowess(5)":0.24},{"Year":1987,"No_Smoothing":0.32,"Lowess(5)":0.27},
{"Year":1988,"No_Smoothing":0.39,"Lowess(5)":0.31},{"Year":1989,"No_Smoothing":0.27,"Lowess(5)":0.33},{"Year":1990,"No_Smoothing":0.45,"Lowess(5)":0.33},{"Year":1991,"No_Smoothing":0.4,"Lowess(5)":0.33},
{"Year":1992,"No_Smoothing":0.22,"Lowess(5)":0.33},{"Year":1993,"No_Smoothing":0.23,"Lowess(5)":0.33},{"Year":1994,"No_Smoothing":0.31,"Lowess(5)":0.34},{"Year":1995,"No_Smoothing":0.44,"Lowess(5)":0.36},
{"Year":1996,"No_Smoothing":0.33,"Lowess(5)":0.4},{"Year":1997,"No_Smoothing":0.46,"Lowess(5)":0.42},{"Year":1998,"No_Smoothing":0.6,"Lowess(5)":0.44},{"Year":1999,"No_Smoothing":0.38,"Lowess(5)":0.47},
{"Year":2000,"No_Smoothing":0.39,"Lowess(5)":0.5},{"Year":2001,"No_Smoothing":0.53,"Lowess(5)":0.52},{"Year":2002,"No_Smoothing":0.62,"Lowess(5)":0.54},{"Year":2003,"No_Smoothing":0.61,"Lowess(5)":0.58},
{"Year":2004,"No_Smoothing":0.53,"Lowess(5)":0.6},{"Year":2005,"No_Smoothing":0.67,"Lowess(5)":0.61},{"Year":2006,"No_Smoothing":0.63,"Lowess(5)":0.62},{"Year":2007,"No_Smoothing":0.66,"Lowess(5)":0.63},
{"Year":2008,"No_Smoothing":0.54,"Lowess(5)":0.64},{"Year":2009,"No_Smoothing":0.65,"Lowess(5)":0.64},{"Year":2010,"No_Smoothing":0.72,"Lowess(5)":0.64},{"Year":2011,"No_Smoothing":0.61,"Lowess(5)":0.66},
{"Year":2012,"No_Smoothing":0.65,"Lowess(5)":0.69},{"Year":2013,"No_Smoothing":0.67,"Lowess(5)":0.74},{"Year":2014,"No_Smoothing":0.74,"Lowess(5)":0.78},{"Year":2015,"No_Smoothing":0.89,"Lowess(5)":0.83},
{"Year":2016,"No_Smoothing":1.01,"Lowess(5)":0.87},{"Year":2017,"No_Smoothing":0.92,"Lowess(5)":0.91},{"Year":2018,"No_Smoothing":0.85,"Lowess(5)":0.93},{"Year":2019,"No_Smoothing":0.97,"Lowess(5)":0.92},
{"Year":2020,"No_Smoothing":1.01,"Lowess(5)":0.91},{"Year":2021,"No_Smoothing":0.84,"Lowess(5)":0.91},{"Year":2022,"No_Smoothing":0.89,"Lowess(5)":0.9}]

图1-1 默认搜索结果

输入语句

```
http://localhost:8000/?start_year=1980&format=xml
```

localhost:8000/?start_year=1900&end_year=1905&format=xml


```
<result>
        <entry>
                <Year>1900</Year>
                <No_Smoothing>-0.08</No_Smoothing>
                <Lowess(5)>-0.20</Lowess(5)>
        </entry>
        <entry>
                <Year>1901</Year>
                <No_Smoothing>-0.16</No_Smoothing>
                <Lowess(5)>-0.24</Lowess(5)>
        </entry>
        <entry>
                <Year>1902</Year>
                <No_Smoothing>-0.28</No_Smoothing>
                <Lowess(5)>-0.26</Lowess(5)>
        </entry>
        <entry>
                <Year>1903</Year>
                <No_Smoothing>-0.37</No_Smoothing>
                <Lowess(5)>-0.28</Lowess(5)>
        </entry>
        <entry>
                <Year>1904</Year>
                <No_Smoothing>-0.48</No_Smoothing>
                <Lowess(5)>-0.31</Lowess(5)>
        </entry>
        <entry>
                <Year>1905</Year>
                <No_Smoothing>-0.26</No_Smoothing>
                <Lowess(5)>-0.34</Lowess(5)>
        </entry>
</result>
```


图1-2 xml搜索结果

输入语句查询从1900到1950，以温度降序的结果，返回csv格式

```
http://localhost:8000/?
start_year=1900&end_year=1950&sort_by=temperature&order=desc&format=csv
```

localhost:8000/?start_year=1900&end_year=1950&sort_by=temperature&order=desc&format=csv

```
1944,0.21,0.08
1941,0.19,0.09
1940,0.13,0.07
1945,0.09,0.04
1943,0.09,0.10
1942,0.07,0.11
1938,0.00,-0.01
1939,-0.02,0.03
1947,-0.02,-0.03
1937,-0.03,-0.06
1946,-0.07,0.01
1900,-0.08,-0.20
1931,-0.09,-0.19
1948,-0.10,-0.07
1949,-0.11,-0.08
1926,-0.11,-0.22
1934,-0.12,-0.15
1915,-0.14,-0.30
1936,-0.14,-0.10
1914,-0.15,-0.31
1901,-0.16,-0.24
1932,-0.16,-0.18
1930,-0.16,-0.19
1950,-0.17,-0.07
1921,-0.19,-0.26
1935,-0.20,-0.14
1928,-0.20,-0.20
1925,-0.22,-0.22
1927,-0.22,-0.21
1906,-0.23,-0.36
1905,-0.26,-0.34
1923,-0.26,-0.24
1924,-0.27,-0.23
1920,-0.27,-0.28
1933,-0.28,-0.17
1919,-0.28,-0.29
1902,-0.28,-0.26
1922,-0.28,-0.25
1918,-0.30,-0.30
1913,-0.34,-0.32
1929,-0.36,-0.19
1916,-0.36,-0.30
1912,-0.36,-0.35
1903,-0.37,-0.28
1907,-0.39,-0.38
1908,-0.43,-0.39
1910,-0.44,-0.41
1911,-0.44,-0.39
1917,-0.46,-0.30
1904,-0.48,-0.31
1909,-0.49,-0.41
```

图1-3 csv搜索结果