第四次上机报告

> 2233316027 王博想

1. 签到统计

问题重述

根据“20250312签到.txt”文件，提取出所有人的签到时间和学号，并进行以下处理：

1. 提取每个学号的唯一签到时间；
2. 从首个签到时间起，以每 5 秒为单位统计每个时间段内的签到人数，绘制变化曲线；
3. 以 37 个“=”为分界符，分别对前后两部分进行相同统计并绘图。

实现思想

- 使用正则表达式提取符合时间格式和 11 位学号的内容；
- 利用 `datetime.strptime` 处理时间字符串，并以首个签到为起点划分 5 秒区间；
- 构造每个时间段内的签到人数统计表；
- 利用 `matplotlib` 绘图展示统计结果；
- 使用分隔符将文件分成两部分分别处理。

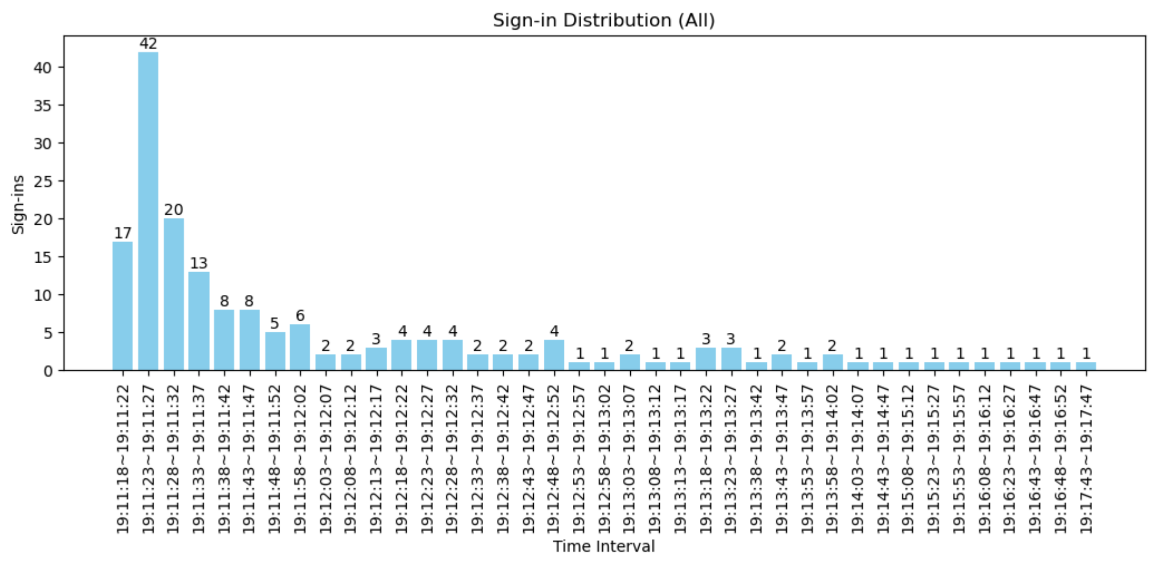
源代码

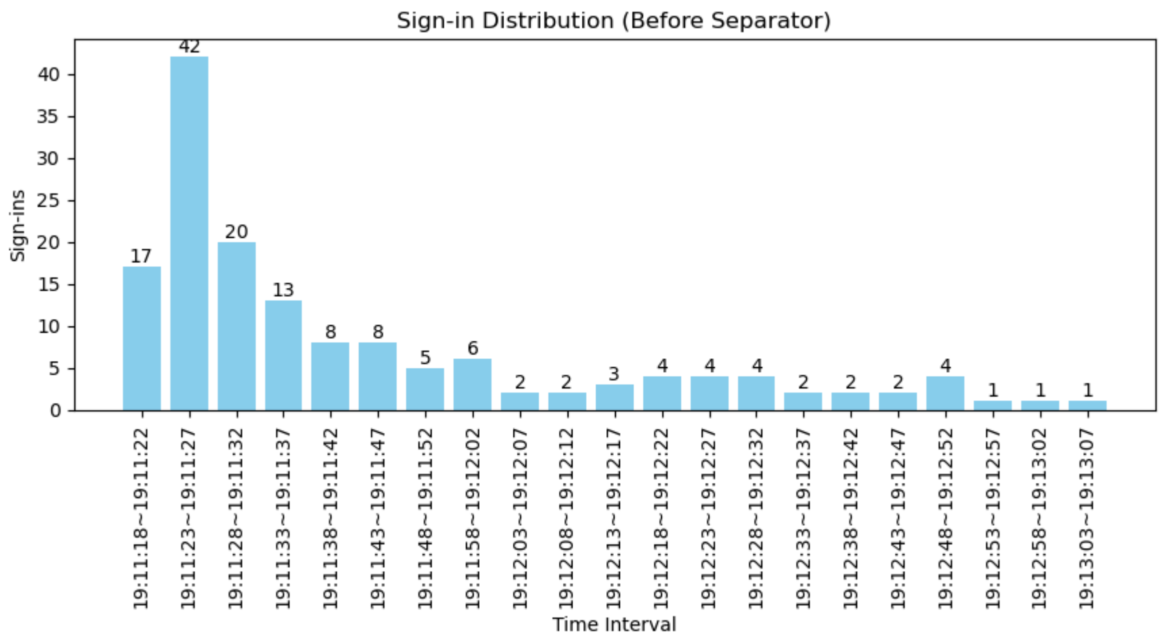
```python
import re
from datetime import datetime, timedelta
import matplotlib.pyplot as plt

with open("20250312签到.txt", "r", encoding="utf-8") as f:
    content = f.read()

content = content.replace('\n', ' ')
parts = content.split("="*37)

def extract_signins(text):
    pattern = re.compile(r"(\d{4}/\d{1,2}/\d{1,2}
\d{1,2}:\d{1,2}:\d{1,2}).*?(\d{10})")
```

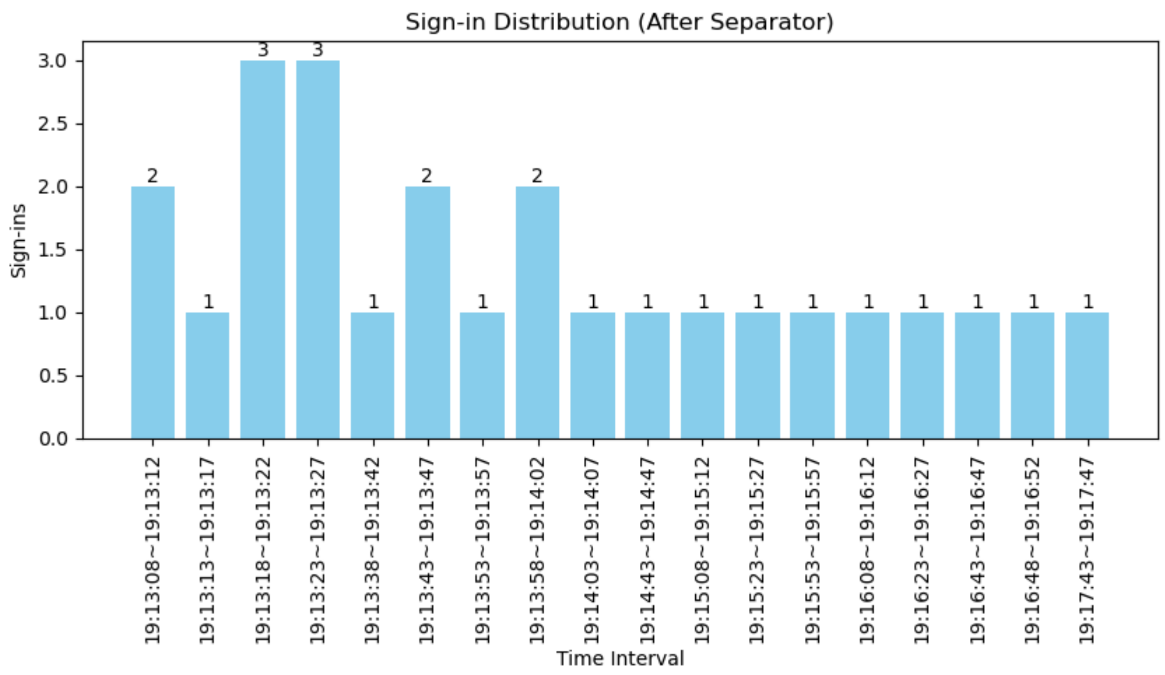
```
    matches = pattern.findall(text)
    signins = {}
    for time_str, sid in matches:
        if sid not in signins:
            signins[sid] = datetime.strptime(time_str, "%Y/%m/%d 
%H:%M:%S")
    return signins

def group_by_interval(signins, interval=5):
    times = list(signins.values())
    if not times:
        return []
    base_time = min(times)
    buckets = {}
    for t in times:
        index = int((t - base_time).total_seconds()) // interval
        start = base_time + timedelta(seconds=index * interval)
        end = start + timedelta(seconds=interval - 1)
        buckets.setdefault((start, end), 0)
        buckets[(start, end)] += 1
    return sorted(buckets.items())

def plot_intervals(intervals, title):
    x = [f"{s.strftime('%H:%M:%S')}~{e.strftime('%H:%M:%S')}" for s, 
e in dict(intervals).keys()]
    y = [v for v in dict(intervals).values()]
    plt.figure(figsize=(10,5))
    bars=plt.bar(x, y, color="skyblue")
    plt.xticks(rotation=90)
    plt.xlabel("Time Interval")
    plt.ylabel("Sign-ins")
    plt.title(title)
    plt.tight_layout()
    for bar in bars:
        height = bar.get_height()
        plt.text(bar.get_x() + bar.get_width()/2., height, 
f'{height}', ha='center', va='bottom')
    plt.show()

signins_all = extract_signins(content)
signins_before = extract_signins(parts[0])
signins_after = extract_signins(parts[1])

intervals_all = group_by_interval(signins_all)
```

```
intervals_before = group_by_interval(signins_before)
intervals_after = group_by_interval(signins_after)

plot_intervals(intervals_all, "Sign-in Distribution (All)")
plot_intervals(intervals_before, "Sign-in Distribution (Before
Separator)")
plot_intervals(intervals_after, "Sign-in Distribution (After
Separator)")
```

结果及说明







图像展示了签到人数随时间变化的趋势：

- 所有人签到人数集中在前几个时间段；
- 分隔符前后的签到分布有明显区别；
- 该方法能够有效分析签到高峰时段。

---

2. 天气预报

问题重述

爬取 http://www.tianqihoubao.com/weather/top/xian.html 网页中一个月内西安的天气数据，完成以下任务：

1. 画出每日最低、最高温度变化曲线，计算平均气温；
2. 统计每种天气状况出现天数，并画出柱状图。

实现思想

- 使用 `requests` 获取网页内容；
- 用 `BeautifulSoup` 和正则表达式提取天气数据；
- 将温度转为整数并进行分析；

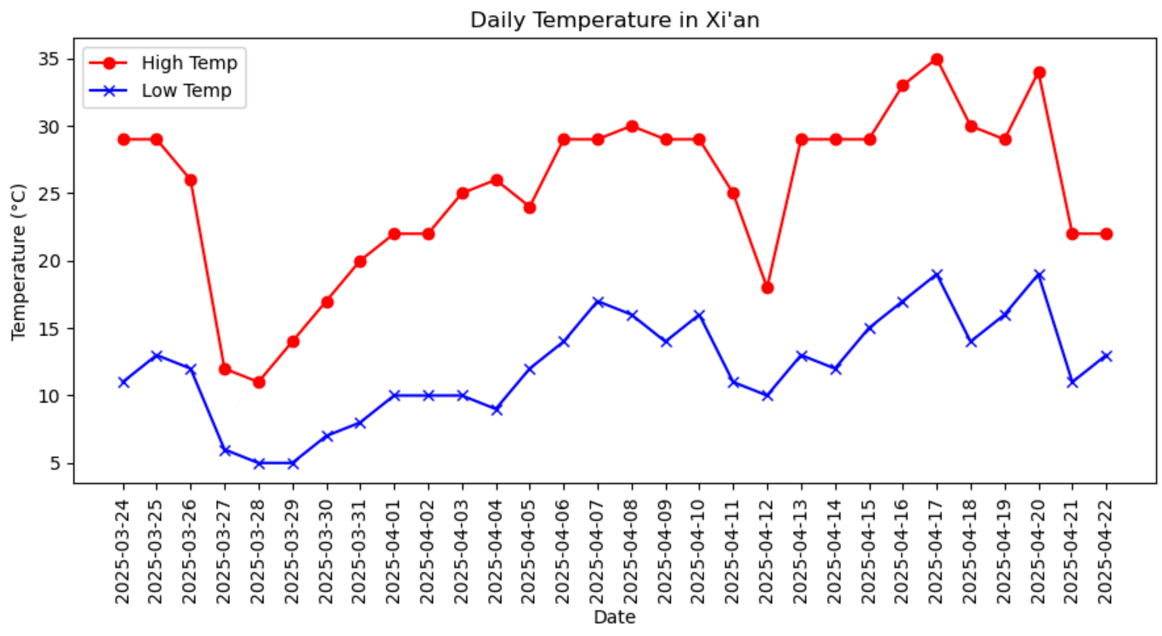
- 用 `matplotlib` 绘制温度变化曲线和天气状况柱状图。

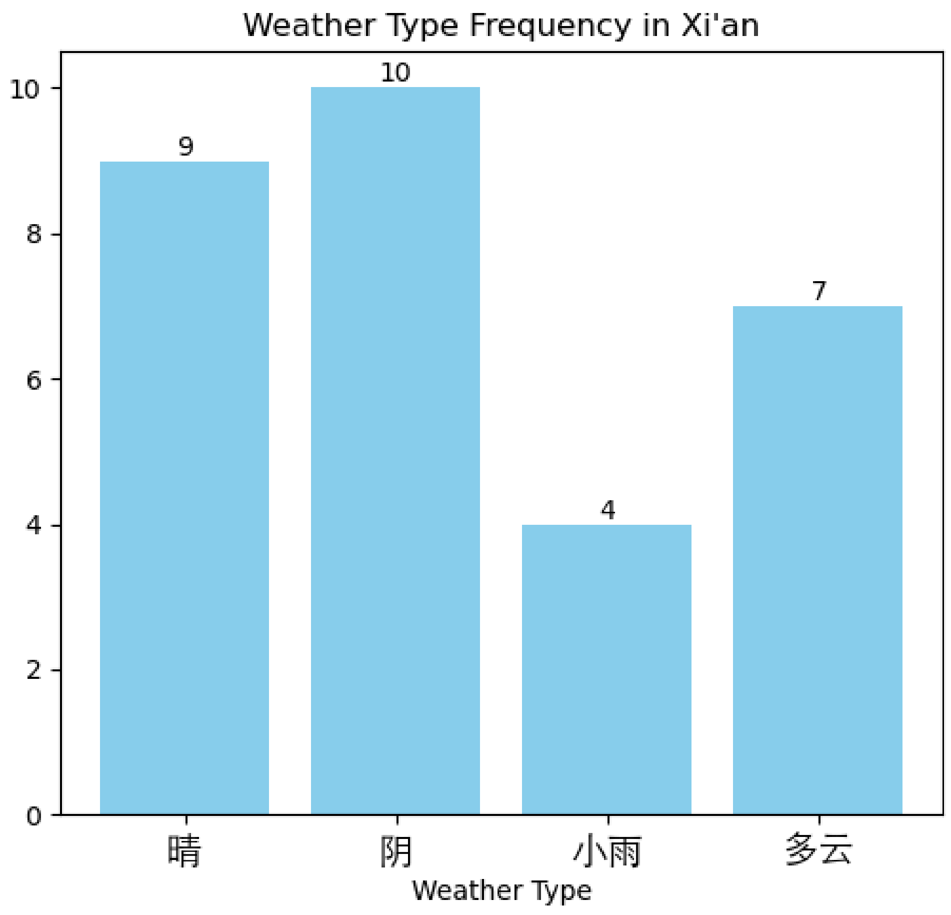
源代码

```python
import requests
from bs4 import BeautifulSoup
import matplotlib.pyplot as plt
import re
from collections import Counter
from matplotlib.font_manager import FontProperties

url = "http://www.tianqihoubao.com/weather/top/xian.html"
headers = {"User-Agent": "Mozilla/5.0"}
response = requests.get(url, headers=headers)
response.encoding = "gb2312"
soup = BeautifulSoup(response.text, "html.parser")

table = soup.find("table")
rows = table.find_all("tr")[2:]
rows.reverse()
dates, highs, lows, weathers_day, weathers_night = [], [], [], [], []

for row in rows:
    cols = row.find_all("td")
    if len(cols) >= 4:
        date = cols[1].text.strip()
        weather_day = cols[2].text.strip()
        weather_night = cols[5].text.strip()
        high = cols[4].text.strip()
        low = cols[7].text.strip()
        high = int(re.findall(r"(-?\d+)", high)[0])
        low = int(re.findall(r"(-?\d+)", low)[0])
        date = re.findall(r"(\d{4}-\d{1,2}-\d{1,2})", date)[0]
        dates.append(date)
        lows.append(low)
        highs.append(high)
        weathers_day.append(weather_day)
        weathers_night.append(weather_night)

avg_low = sum(lows) / len(lows)
avg_high = sum(highs) / len(highs)

print("Average High Temperature:", avg_high)
print("Average Low Temperature:", avg_low)

plt.figure(figsize=(10, 5))
plt.plot(dates, highs, label="High Temp", color="red", marker="o")
plt.plot(dates, lows, label="Low Temp", color="blue", marker="x")
plt.xticks(rotation=90)
plt.xlabel("Date")
plt.ylabel("Temperature (°C)")
plt.title("Daily Temperature in Xi'an")
plt.legend()
plt.tight_layout()
plt.show()

weather_counter = Counter(weathers_day)

font = FontProperties(fname="./STHeiti Light.ttc", size=14)
plt.figure(figsize=(10, 5))
bars = plt.bar(weather_counter.keys(), weather_counter.values(),
color="skyblue")
plt.xticks(rotation=0, fontproperties=font)
plt.xlabel("Weather Type")
plt.ylabel("Days")
plt.title("Weather Type Frequency in Xi'an")
plt.tight_layout()
for bar in bars:
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2., height, f'{height}',
ha='center', va='bottom')
plt.show()
```

结果及说明

```
Average High Temperature: 25.266666666666666
Average Low Temperature: 12.166666666666666
```





- 曲线图展示了温度在一个月内的波动趋势；

- 平均气温结果揭示出西安春季气候特点；
- 柱状图显示晴、阴、雨等天气的出现频率，有助于了解月内天气分布。