**大连财经学院**

**《数据采集与数据清洗》实训报告**

**实训名称 \_数据采集器应用领域实训 \_**

**专 业： \_ \_\_ \_**

**年级班级： \_ \_**

**学生姓名： \_ \_**

**学 号： \_**

**指导教师： \_石佳鑫 \_**

**实训日期： 2024年11月15日 \_**

**实训地点： \_D204 \_**

**实训成绩： \_ \_**

**大连财经学院教务处制**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **实训形式** | **非独立设课实训** | | | √ | **独立设课实训** |  | | **集中性综合实训** |  |
| **实训目的** | 通过实际工作中的真实案例，让学生熟悉各类数据采集领域典型网站，在实训中获得数据采集能力。通过上机实训，强化学生操作方面的感性认识，明确操作的具体要求，有利于学生锻炼动手操作能力，培养工匠精神。 | | | | | | | | |
| **实训环境与设备** | 1.计算机一台；  2.Win10操作系统；  3.PyCharm；  4.word。 | | | | | | | | |
| **实训内容** | 通过正则表达式爬取大连市未来7天的天气情况，包括日期、温度、风力等数据，爬取的具体网站无限制。 | | | | | | | | |
| **实训步骤** | 1. 发送请求获取网页内容：   使用 requests.get(url) 向指定网址（大连市天气预报页面）发送请求，获取网页的 HTML 内容。   1. 设置网页编码：   使用 response.encoding = 'utf-8' 确保网页内容按正确的 UTF-8 编码解析。   1. 使用正则表达式提取数据：   通过正则表达式提取所需的天气信息：  日期：提取每一天的日期（如“周二”）。  天气情况：提取天气状况（如“晴”或“多云”）。  温度：提取白天气温和夜间气温（如“6℃ / 0℃”）。  风力：提取风力信息（如“西风”）。   1. 解析并保存数据：   使用 re.findall() 方法从网页内容中提取所有匹配的数据，保存到相应的列表中。   1. 输出结果：   遍历提取的数据，按日期、天气情况、温度和风力信息格式化输出。 | | | | | | | | |
| **结果与分析** | import requests  import re  # 目标网址（大连市未来7天天气）  url = "https://www.weather.com.cn/weather15d/101070201.shtml"  # 发送请求获取网页内容，并确保使用UTF-8编码  response = requests.get(url)  response.encoding = 'utf-8' # 明确设置编码为UTF-8  if response.status\_code == 200:  html\_content = response.text  else:  print(f"请求失败，状态码：{response.status\_code}")  exit()  # 使用正则表达式提取天气信息  # 1. 提取日期  dates\_pattern = r'<span class="time">(.\*?)</span>'  dates = re.findall(dates\_pattern, html\_content)  # 2. 提取天气状况  weather\_pattern = r'<span class="wea">(.\*?)</span>'  weather\_conditions = re.findall(weather\_pattern, html\_content)  # 3. 提取温度（白天/夜间温度）  temperature\_pattern = r'<span class="tem"><em>(.\*?)</em>/(.\*?)</span>'  temperatures = re.findall(temperature\_pattern, html\_content)  # 4. 提取风力  wind\_pattern = r'<span class="wind">(.\*?)</span>'  winds = re.findall(wind\_pattern, html\_content)  # 输出提取的数据  for i in range(len(dates)):  print(f"日期: {dates[i]}")  print(f"天气情况: {weather\_conditions[i]}")  print(f"白天气温: {temperatures[i][0]} | 夜间气温: {temperatures[i][1]}")  print(f"风力: {winds[i]}")  print('-' \* 40)  程序输出： 日期: 周二（24日）  天气情况: 晴  白天气温: 6℃ | 夜间气温: 0℃  风力: 西风  ----------------------------------------  日期: 周三（25日）  天气情况: 晴  白天气温: 4℃ | 夜间气温: -1℃  风力: 西风转北风  ----------------------------------------  日期: 周四（26日）  天气情况: 晴  白天气温: -1℃ | 夜间气温: -4℃  风力: 西北风转北风  ----------------------------------------  日期: 周五（27日）  天气情况: 雪转多云  白天气温: 0℃ | 夜间气温: -3℃  风力: 西北风  ----------------------------------------  日期: 周六（28日）  天气情况: 晴转多云  白天气温: 2℃ | 夜间气温: -2℃  风力: 西风转北风  ----------------------------------------  日期: 周日（29日）  天气情况: 多云  白天气温: -2℃ | 夜间气温: -5℃  风力: 北风  ----------------------------------------  日期: 周一（30日）  天气情况: 晴转阴  白天气温: -4℃ | 夜间气温: -6℃  风力: 北风  ----------------------------------------  日期: 周二（31日）  天气情况: 阴  白天气温: 0℃ | 夜间气温: -5℃  风力: 西风  ---------------------------------------- | | | | | | | | |
| **实训体会** | 通过本次实训，我深入理解了网页数据爬取的流程和方法。实践中，学习了如何使用 Python 的 `requests` 库获取网页内容，并通过正则表达式提取所需的具体信息。在处理字符编码时遇到了一些问题，但通过设置正确的编码方式解决了乱码问题。总的来说，本次实训加深了我对数据爬取技术的理解，并提升了我在实际应用中的问题解决能力。 | | | | | | | | |
| **实训成绩** | | **正确性**  **（60分）** | **规范性**  **（20分）** | | **及时性**  **（10分）** | | **实训纪律**  **（10分）** | | |
|  |  | |  | |  | | |
| **指导老师签字** | | **石佳鑫** | | | | | | | |