**大连财经学院**

**《数据采集与数据清洗》实训报告**

**实训名称 \_XPath语言应用实训 \_**

**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1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** | 通过实际生活中的真实案例，让学生了解数据采集的基本功能和使用方法。通过上机实验，强化学生操作方面的感性认识，明确操作的具体要求，提升动手操作能力。通过本次实验有利于学生将所学内容应用于以后的学习与工作当中，从而培养职业素养和社会责任心。 | | | | | | | | |
| **实训环境与设备** | 1.计算机一台；  2.Win10操作系统；  3.PyCharm；  4.截图工具。 | | | | | | | | |
| **实训内容** | 1.爬取百度新闻网页源代码。网址：https://news.baidu.com/。  2.爬取一张与大连财经学院有关的图片，命名为学号+姓名。  3.爬取博客园前10页新闻标题，网址：https://news.cnblogs.com/。 | | | | | | | | |
| **实训步骤** | 1. 百度源代码 2. 发送 HTTP 请求：   使用 requests.get() 方法向百度新闻的网址（https://news.baidu.com/）发送 GET 请求，获取网页源代码。   1. 检查响应状态：   使用 response.status\_code 检查请求是否成功。如果状态码为 200，说明请求成功，继续处理数据。   1. 保存网页源代码：   使用 response.text 获取网页的 HTML 内容，并将其保存为本地文件，如 baidu\_news.html。   1. 爬取图片 2. 构造百度图片搜索 URL：   构建百度图片搜索的 URL，查询关键词为“大连财经学院”：   1. 获取页面源代码：   使用 requests.get(url) 获取网页的 HTML 内容。   1. 解析 HTML 内容：   使用 BeautifulSoup 解析获取的 HTML 内容，并找到第一个 <img> 标签的 src 属性，提取出图片的 URL。   1. 下载图片：   使用 requests.get() 获取图片的二进制数据，并保存为文件。   1. 爬取新闻标题 2. 构造博客园新闻页面 URL：   博客园新闻页面的 URL 格式为 https://news.cnblogs.com/n/next\_page/{page}/，其中 {page} 是页面编号。  使用循环爬取前 10 页的新闻标题。   1. 发送 HTTP 请求并获取页面内容：   对每一页（从 1 到 10）发送 GET 请求，获取页面的 HTML 内容。   1. 解析 HTML 内容：   使用 BeautifulSoup 解析每页的 HTML 内容，查找所有新闻标题，定位到 class="news\_entry\_title" 的 <a> 标签。   1. 提取新闻标题：   使用 .get\_text(strip=True) 方法提取新闻标题，去除多余的空格。 | | | | | | | | |
| **结果与分析** | import requests  # 目标网址  url = "https://news.baidu.com/"  # 获取网页源代码  response = requests.get(url)  # 检查请求是否成功  if response.status\_code == 200:  # 输出网页源代码  with open("baidu\_news.html", "w", encoding="utf-8") as f:  f.write(response.text)  print("百度新闻网页源代码已保存。")  else:  print(f"请求失败，状态码：{response.status\_code}")    from selenium import webdriver  from selenium.webdriver.edge.service import Service  from selenium.webdriver.common.by import By  import time  import requests  from webdriver\_manager.microsoft import EdgeChromiumDriverManager  url = "https://image.baidu.com/search/index?tn=baiduimage&word=大连财经学院"  # 使用 WebDriverManager 获取 Edge WebDriver  driver = webdriver.Edge(service=Service(EdgeChromiumDriverManager().install()))  # 启动浏览器，访问目标页面  driver.get(url)  # 等待页面加载  time.sleep(5) # 可以增加延时等待页面加载完全  # 查找所有图片元素  img\_elements = driver.find\_elements(By.TAG\_NAME, "img")  img\_urls = [img.get\_attribute("src") for img in img\_elements if img.get\_attribute("src")]  if img\_urls:  # 选择第几张图片  img\_url = img\_urls[30]  print(f"找到图片: {img\_url}")  # 使用 requests 下载图片  img\_data = requests.get(img\_url).content  with open("img.jpg", "wb") as f:  f.write(img\_data)  print("图片已保存。")  else:  print("未找到图片。")  # 关闭浏览器  driver.quit()    import requests  from bs4 import BeautifulSoup  # 存储所有新闻标题的列表  all\_titles = []  # 爬取前10页  for page in range(1, 11):  url = f"https://news.cnblogs.com/n/page/{page}/"  # 获取网页内容  response = requests.get(url)  if response.status\_code == 200:  soup = BeautifulSoup(response.text, "html.parser")  # 找到所有新闻标题  titles = soup.find\_all("h2", class\_="news\_entry")  for title in titles:  all\_titles.append(title.get\_text(strip=True))  else:  print(f"请求失败，状态码：{response.status\_code}")  # 打印前10页新闻标题  for i, title in enumerate(all\_titles, start=1):  print(f"{i}. {title}") | | | | | | | | |
| **实训体会** | 在本次实训过程中，我深入学习了如何使用 Python 进行网页数据的爬取。通过实际操作，我掌握了基本的爬虫技术，并应用到不同的任务中，如爬取网页源代码、图片下载以及从多个页面中提取数据。以下是我在实训过程中的一些感悟：  对爬虫技术的理解更加深入： 通过完成这次实训，我对网页数据爬取的原理有了更加深刻的理解。从使用 requests 获取网页源代码，到使用 BeautifulSoup 解析 HTML 内容，每一步都加深了我对爬虫技术的认知。特别是对动态网页的理解，虽然这次实训主要集中在静态网页上，但也让我意识到处理动态内容时需要用到更多的工具，如 Selenium。  数据提取和解析是关键： 在爬取网页时，最关键的一步是如何从网页源代码中提取到需要的数据。通过使用 BeautifulSoup 和正则表达式，我学会了如何灵活地解析 HTML 内容，并获取目标数据。这对于后续的爬虫项目和数据分析工作都有很大的帮助。  网页结构的变化挑战爬虫稳定性： 在进行任务时，我发现不同网站的结构各异，甚至同一网站的页面结构也会发生变化，这使得爬虫的稳定性受到了挑战。为了提高爬虫的鲁棒性，我学习了如何通过检查网页元素的类名、标签等特征来定位所需的内容，并避免因网页结构的微小变化导致程序崩溃。  数据存储和管理的重要性： 在抓取数据后，如何有效地存储和管理这些数据也成为一个重要的课题。尽管这次实训并没有涉及到数据库的使用，但我意识到数据的存储形式应该根据实际需求来选择，例如将数据存储为 CSV 文件、数据库或其他格式。  问题解决能力的提升： 在整个实训过程中，我遇到了很多问题，例如网页请求失败、图片链接无效、HTML 解析错误等。每遇到问题，我都通过查阅文档、搜索资料和调试代码进行解决，这大大提升了我的问题分析和解决能力。  总的来说，这次实训让我更加熟悉了 Python 爬虫技术，并对如何从网络上提取和处理数据有了更深入的理解。这不仅为我日后的数据分析和处理奠定了基础，也让我更加了解了爬虫技术在实际应用中的挑战和解决方案。 | | | | | | | | |
| **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** | | | | | | | |