# 东莞理工学院网络空间安全学院

# 实验报告

**课程名称：**Python 数据分析与应用　　**学期：**2023 年秋季

| 实验名称 | 网络爬虫 | | | 实验序号 | 2 |
|---|---|---|---|---|---|
| 姓　　名 | 皮昊旋 | 学　　号 | 2021428010127 | 班　　级 | 21 杨班 |
| 实验地点 | 8B403 | 实验日期 | 20231018 | 指导老师 | 丁烨 |
| 教师评语 | - | | | 实验成绩 | - |
| | | | | 百分制 | 100 |
| 同组同学 | 无 | | | | |

一、　实验目标

1.1　了解爬虫机制；

1.2　熟悉爬虫工具及代码；

1.3　使用爬虫工具，部署爬虫代码。

二、　实验条件

2.1　硬件条件：PC 兼容机或 Mac；

2.2　软件条件：Python、PyCharm 或 VS Code 等 IDE。

三、　实验内容

3.1　安装 Scrapy 或其他爬虫，或开启 SaaS 爬虫服务；

3.2　爬取名人名言：http://quotes.toscrape.com/；

3.3　加入分页识别机制；

3.4　爬取名人名言作者信息。

四、　实验作业及分析

4.1　实验过程

① 使用 pip install -U scrapy 安装相关框架

```
PS E:\python-data-analyse> pip install -U scrapy
Collecting scrapy
  Obtaining dependency information for scrapy from https://files.pythonhosted.org/packages/08/66/22ed9609df4b6d9
etadata
  Downloading Scrapy-2.11.0-py2.py3-none-any.whl.metadata (5.2 kB)
Collecting Twisted<23.8.0,>=18.9.0 (from scrapy)
  Downloading Twisted-22.10.0-py3-none-any.whl (3.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.1/3.1 MB 2.9 MB/s eta 0:00:00
Collecting cryptography>=36.0.0 (from scrapy)
  Obtaining dependency information for cryptography>=36.0.0 from https://files.pythonhosted.org/packages/d7/78/2
37-abi3-win_amd64.whl.metadata
  Downloading cryptography-41.0.4-cp37-abi3-win_amd64.whl.metadata (5.3 kB)
Collecting cssselect>=0.9.1 (from scrapy)
  Downloading cssselect-1.2.0-py2.py3-none-any.whl (18 kB)
Collecting itemloaders>=1.0.1 (from scrapy)
  Downloading itemloaders-1.1.0-py3-none-any.whl (11 kB)
Collecting parsel>=1.5.0 (from scrapy)
  Downloading parsel-1.8.1-py2.py3-none-any.whl (17 kB)
```

② 编写爬虫代码

```python
from typing import Any, Iterable
import scrapy
from scrapy.http import Request, Response

class QuotesSpider(scrapy.Spider):
    name = "quotes"

    def start_requests(self):
        urls = ['http://quotes.toscrape.com/page/1','http://quotes.toscrape.com/page/2']
        for url in urls:
            yield scrapy.Request(url=url, callback=self.parse)

    def parse_author_info(self, response):
        yield {
            'about': response.css('div.author-description::text').get()
        }

    def parse(self, response):
        next_page = response.css('li.next a::attr(href)').get()
        if next_page is not None:
            yield response.follow(next_page, self.parse)

        for quote in response.css("div.quote"):
            author_info_page = 'http://quotes.toscrape.com' + quote.css('div.quote a::attr(href)').get()
            yield Request(author_info_page, callback=self.parse_author_info, meta={'quote': quote})

        page = response.url.split("/")[-2]
        filename = 'quotes-%s.html' %page
        with open(filename,"wb") as f:
            f.write(response.body)
        self.log("Saved file %s" % filename)

        for quote in response.css('div.quote'):
            author_info_page = 'http://quotes.toscrape.com' + quote.css('div.quote a::attr(href)').get()
            yield Request(author_info_page, callback=self.parse_author_info, meta={'quote': quote})

        page = response.url.split("/")[-2]
        filename = 'quotes-%s.html' % page
        with open(filename,"wb") as f:
            f.write(response.body)
        self.log("Saved file %s" % filename)

    def parse_author_info(self, response):
        quote = response.meta['quote']
        yield {
            'text' : quote.css('span.text::text').get(),
            'author' : quote.css('small.author::text').get(),
            'tags' : quote.css('div.tags a.tag::text').getall(),
            'about' : response.css('div.author-description::text').get(),
        }

```

为了爬取作者详细信息，加入新函数

```python
def parse_author_info(self, response):
    yield {
        'about': response.css('div.author-description::text').get()
    }
```

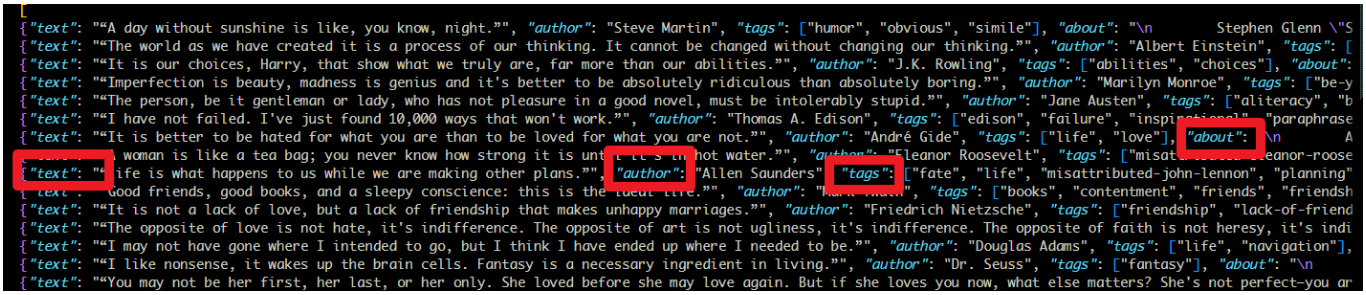
专门获取作者页的信息

因为获取作者页的信息也需要 请求一次，所以在组织信息前用 meta 来记录请求间的关系。

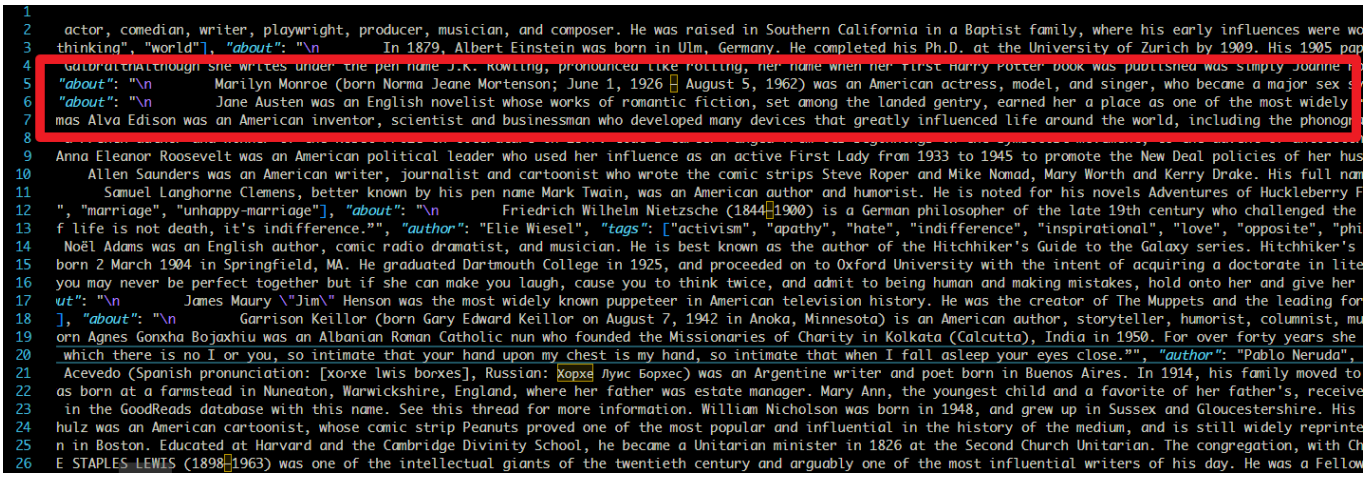
③ 测试并完善爬虫代码

```
'finish_time': datetime.datetime(2023, 10, 18, 1, 38, 26, 144025, tzinfo=datetime.timezone.utc),
'item_scraped_count': 20,
'log_count/DEBUG': 30,
'log_count/INFO': 11,
'response_received_count': 3,
'robotstxt/request_count': 1,
'robotstxt/response_count': 1,
'robotstxt/response_status_count/404': 1,
'scheduler/dequeued': 4,
'scheduler/dequeued/memory': 4,
'scheduler/enqueued': 4,
'scheduler/enqueued/memory': 4,
'start_time': datetime.datetime(2023, 10, 18, 1, 38, 23, 944890, tzinfo=datetime.timezone.utc)}
2023-10-18 09:38:26 [scrapy.core.engine] INFO: Spider closed (finished)
PS E:\python-data-analyse\lab2\tutorial>
```

4.2 实验结果




About 内容是作者页的详细内容



五、 实验总结

本次实验使用 Scrapy 框架编写了一个爬虫程序，从 http://quotes.toscrape.com 上爬取名言和作者信息，并实现了将名、作者、tags 和作者简介进行保存的功能。

在实验过程中，首先定义了一个 QuotesSpider 的 Spider 类。在 start_requests 方法中，设置了要爬取的页面 URL 列表，并通过循环为每个 URL 发起请求，指定回调函数为 parse 方法。

在 parse 方法中，先获取下一页的链接，如果存在下一页，则继续发起请求。然后，利用 CSS 选择器提取出每个名言的文本、作者、标签等信息，并通过 yield 语句返回一个包含这些信息的字典。然后对每个名言提取到的链接发起请求，以获取作者信息页面的内容。

在新的回调函数 parse_author_info 中，通过 response.meta 获取到之前存入的名言对象，然后从响应中提取作者简介信息，并将其添加到之前的名言字典中。最后，再次通过 yield 语句返回完整的名言信息。

本次实验通过 Scrapy 框架实现了对名言网站的爬取，提取了所需的名言和作者信息，并将作者简介作为名言的一部分进行保存。通过这次实验，我对 Scrapy 的使用有了更深入的了解，并掌握了如何在爬虫程序中处理多个请求的数据关系，以及如何将爬取到的数据进行保存和后续处理。