USTC-AD/2024 课程作业 实验报告

实验 2	**DBLP** 论文信息获取与整理
马天开	PB21000030
Due: 2024.03.28	Submitted: 2024.03.14

In [ ]:
```
import re
import requests
import json
from pprint import pprint
```

模拟浏览器行为。相关 raw Headers 可直接从 Firefox 复制。

In [ ]:
```
headers_raw = """
Host: dblp.uni-trier.de
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:123.0) Gecko
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=
Accept-Encoding: gzip, deflate, br
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Cookie: dblp-search-mode=c; dblp-dismiss-new-feature-2022-01-27=1
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: cross-site
"""

headers = {}
for line in headers_raw.split("\n"):
    if line.strip() == "":
        continue
    k, v = line.split(": ")
    headers[k] = v

print(headers)
```

```
{'Host': 'dblp.uni-trier.de', 'User-Agent': 'Mozilla/5.0 (Macintosh; Intel
Mac OS X 10.15; rv:123.0) Gecko/20100101 Firefox/123.0', 'Accept': 'text/h
tml,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/
*;q=0.8', 'Accept-Language': 'zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-U
S;q=0.3,en;q=0.2', 'Accept-Encoding': 'gzip, deflate, br', 'DNT': '1', 'Se
c-GPC': '1', 'Connection': 'keep-alive', 'Cookie': 'dblp-search-mode=c; db
lp-dismiss-new-feature-2022-01-27=1', 'Upgrade-Insecure-Requests': '1', 'S
ec-Fetch-Dest': 'document', 'Sec-Fetch-Mode': 'navigate', 'Sec-Fetch-Sit
e': 'cross-site'}
```

In [ ]:
```
# open https://dblp.uni-trier.de/db/conf/kdd/kdd2023.html and save to pag
req = requests.get("https://dblp.uni-trier.de/db/conf/kdd/kdd2023.html",
with open("page.txt", "w") as f:
    f.write(req.text)
```

提供再次读取的入口，以便于检查：

In [ ]:
```
with open("page.txt", "r") as f:
    page = f.read()
```

以下定义仅供参考，Python 内置的 json 模块调用反射的方式存在问题，后续以 dict 的方式存储。

In [ ]:
```
# this is pratically useless since builtin reflection in python is bullsh
# using simple dict + list instead
class Paper:
    authors: list[str]
    title: str
    startPage: int
    endPage: int

    def __init__(
        self,
        authors: list[str] = [],
        title: str = "",
        startPage: int = 0,
        endPage: int = 0
    ):
        self.authors = authors
        self.title = title
        self.startPage = startPage
        self.endPage = endPage


class Track:
    track: str
    papers: list[Paper]

    def __init__(
        self,
        track: str = "",
        papers: list[Paper] = []
    ):
        self.track = track
        self.papers = papers


tracks = []
```

用于 parse HTML 的 Regex， 正常情况下应先 parse DOM 结构再用 selector，但内置库无对应方法。

In [ ]:
```
# regex patterns:
# since no DOM parsing package is built-in, we have to use regex to parse

track_full_pattern = re.compile(r'<header class="h2">(.*?)<meta property=
track_name_pattern = re.compile(r'<h2 id="(.+?)">(.+?)</h2>')
paper_full_pattern = re.compile(r'<li class="entry inproceedings"(.*?)<me
author_name_pattern = re.compile(r'<span itemprop="name" title=".*?">(.*?
title_pattern = re.compile(r'<span class="title" itemprop="name">(.+?)</s
pagination_pattern = re.compile(r'<span itemprop="pagination">(.+?)</span
```

上述 Regex 中每一步匹配都是非贪婪的，因此结尾使用了更多的特征，同时匹配结果是无交的，保证后续代码的简洁。

下面的代码嵌套使用 Regex 获取的内容，保证结构的一致性（不会出现一篇文章出现在另一个分类下的问题）。

In [ ]:
```
tracks = []
for track_raw in track_full_pattern.findall(page)[:2]:
    track = {}
    track["track"] = track_name_pattern.search(track_raw).group(2)
    track["papers"] = []

    for paper_raw in paper_full_pattern.findall(track_raw):
        paper = {}
        authors = []
        for author in author_name_pattern.findall(paper_raw):
            authors.append(author)
        paper["authors"] = authors
        paper["title"] = title_pattern.search(paper_raw).group(1)

        pagination = pagination_pattern.search(paper_raw).group(1)
        if '-' in pagination:
            _pagination = pagination.split('-')
            paper["startPage"] = int(_pagination[0])
            paper["endPage"] = int(_pagination[1])
        else:
            paper["startPage"] = int(pagination)
            paper["endPage"] = int(pagination)
        track["papers"].append(paper)

        # print(paper.authors)

    print(track["track"], "\n", len(track["papers"]))

    tracks.append(track)
```

```
Research Track Full Papers
 312
Applied Data Track Full Papers
 182
```

依照要求保存文件

In [ ]:
```
# save tracks to json
with open("kdd23.json", "w") as f:
    f.write(json.dumps(tracks, indent=2))
```

获取前2个分类中，每个分类前10个作者的url：

In [ ]:
```
url_pattern = re.compile(r'https://dblp.uni-trier.de/pid/\d+/\d+.html')

author_urls = []
for track_raw in track_full_pattern.findall(page)[:2]:
    for paper_raw in paper_full_pattern.findall(track_raw)[:10]:
        for author in author_name_pattern.findall(paper_raw):
            url = url_pattern.search(paper_raw).group(0)
            author_urls.append(url)
```

['https://dblp.uni-trier.de/pid/211/5760.html', 'https://dblp.uni-trier.de/pid/211/5760.html', 'https://dblp.uni-trier.de/pid/211/5760.html', 'https://dblp.uni-trier.de/pid/247/9288.html', 'https://dblp.uni-trier.de/pid/247/9288.html', 'https://dblp.uni-trier.de/pid/247/9288.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/221/2843.html', 'https://dblp.uni-trier.de/pid/201/1957.html', 'https://dblp.uni-trier.de/pid/201/1957.html', 'https://dblp.uni-trier.de/pid/201/1957.html', 'https://dblp.uni-trier.de/pid/201/1957.html', 'https://dblp.uni-trier.de/pid/270/0786.html', 'https://dblp.uni-trier.de/pid/270/0786.html', 'https://dblp.uni-trier.de/pid/270/0786.html', 'https://dblp.uni-trier.de/pid/270/0786.html', 'https://dblp.uni-trier.de/pid/270/0786.html', 'https://dblp.uni-trier.de/pid/250/9281.html', 'https://dblp.uni-trier.de/pid/250/9281.html', 'https://dblp.uni-trier.de/pid/250/9281.html', 'https://dblp.uni-trier.de/pid/250/9281.html', 'https://dblp.uni-trier.de/pid/250/9281.html', 'https://dblp.uni-trier.de/pid/250/9281.html', 'https://dblp.uni-trier.de/pid/300/4385.html', 'https://dblp.uni-trier.de/pid/300/4385.html', 'https://dblp.uni-trier.de/pid/300/4385.html', 'https://dblp.uni-trier.de/pid/300/4385.html', 'https://dblp.uni-trier.de/pid/222/8125.html', 'https://dblp.uni-trier.de/pid/222/8125.html', 'https://dblp.uni-trier.de/pid/222/8125.html', 'https://dblp.uni-trier.de/pid/222/8125.html', 'https://dblp.uni-trier.de/pid/222/8125.html', 'https://dblp.uni-trier.de/pid/222/8125.html', 'https://dblp.uni-trier.de/pid/169/9751.html', 'https://dblp.uni-trier.de/pid/169/9751.html', 'https://dblp.uni-trier.de/pid/169/9751.html', 'https://dblp.uni-trier.de/pid/169/9751.html', 'https://dblp.uni-trier.de/pid/169/9751.html', 'https://dblp.uni-trier.de/pid/169/9751.html', 'https://dblp.uni-trier.de/pid/224/0256.html', 'https://dblp.uni-trier.de/pid/224/0256.html', 'https://dblp.uni-trier.de/pid/224/0256.html', 'https://dblp.uni-trier.de/pid/224/0256.html', 'https://dblp.uni-trier.de/pid/79/8611.html', 'https://dblp.uni-trier.de/pid/79/8611.html', 'https://dblp.uni-trier.de/pid/79/8611.html', 'https://dblp.uni-trier.de/pid/79/8611.html', 'https://dblp.uni-trier.de/pid/326/4004.html', 'https://dblp.uni-trier.de/pid/326/4004.html', 'https://dblp.uni-trier.de/pid/326/4004.html', 'https://dblp.uni-trier.de/pid/326/4004.html', 'https://dblp.uni-trier.de/pid/275/9924.html', 'https://dblp.uni-trier.de/pid/275/9924.html', 'https://dblp.uni-trier.de/pid/275/9924.html', 'https://dblp.uni-trier.de/pid/161/9645.html', 'https://dblp.uni-trier.de/pid/161/9645.html', 'https://dblp.uni-trier.de/pid/161/9645.html', 'https://dblp.uni-trier.de/pid/161/9645.html', 'https://dblp.uni-trier.de/pid/161/9645.html', 'https://dblp.uni-trier.de/pid/339/7339.html', 'https://dblp.uni-trier.de/pid/339/7339.html', 'https://dblp.uni-trier.de/pid/339/7339.html', 'https://dblp.uni-trier.de/pid/339/7339.html', 'https://dblp.uni-trier.de/pid/339/7339.html', 'https://dblp.uni-trier.de/pid/339/7339.html', 'https://dblp.uni-trier.de/pid/339/7339.html', 'https://dblp.uni-trier.de/pid/339/7339.html', 'https://dblp.uni-trier.de/pid/259/2365.html', 'https://dblp.uni-trier.de/pid/259/2365.html', 'https://dblp.uni-trier.de/pid/259/2365.html', 'https://dblp.uni-trier.de/pid/224/5573.html', 'https://dblp.uni-trier.de/pid/224/5573.html', 'https://dblp.uni-trier.de/pid/224/5573.html', 'https://dblp.uni-trier.de/pid/224/5573.html', 'https://dblp.uni-trier.de/pid/224/5573.html', 'https://dblp.uni-trier.de/pid/224/5573.html', 'https://dblp.uni-trier.de/pid/224/5573.html', 'https://dblp.uni-trier.de/pid/224/5573.html', 'https://dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.uni-trier.de/pid/264/5777.html', 'https://

```
dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.uni-trier.de/pid/264/5
777.html', 'https://dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.un
i-trier.de/pid/264/5777.html', 'https://dblp.uni-trier.de/pid/264/5777.htm
l', 'https://dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.uni-trie
r.de/pid/264/5777.html', 'https://dblp.uni-trier.de/pid/264/5777.html', 'h
ttps://dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.uni-trier.de/pi
d/264/5777.html', 'https://dblp.uni-trier.de/pid/264/5777.html', 'https://
dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.uni-trier.de/pid/264/5
777.html', 'https://dblp.uni-trier.de/pid/264/5777.html', 'https://dblp.un
i-trier.de/pid/15/10240.html', 'https://dblp.uni-trier.de/pid/15/10240.htm
l', 'https://dblp.uni-trier.de/pid/15/10240.html', 'https://dblp.uni-trie
r.de/pid/15/10240.html']
```

从每个作者的url中获取作者的相关信息

在实践中发现了三种不同标记文章发表期刊、时间的标记方式，用 try-except 分别匹配对应模式。

每个页面中 orcID 可能包含其他作者，采用完整的 <a/> 标签匹配，如果匹配不到则标记为空字符串。

In [ ]:

```python
name_pattern = re.compile(r'<span class="name primary" itemprop="name">(.
orcid_pattern = re.compile(r'<a href="https://orcid.org/([0-9]{4}-[0-9]{4
paper_pattern = re.compile(r'<li class="entry inproceedings toc"(.*?)<met

publish_info_pattern_A = re.compile(r'<span itemprop="isPartOf" itemscope
publish_info_pattern_B = re.compile(r'<span itemprop="name">(.*?)</span><
publish_info_pattern_C = re.compile(r'<span itemprop="name">(.*?)</span><

def handle_url(url: str):
    text = requests.get(url, headers=headers).text

    name = name_pattern.search(text).group(1)
    try:
        orcid = orcid_pattern.search(text).group(1)
    except:
        orcid = ""

    papers = []
    for paper_raw in paper_pattern.findall(text):
        authors = []
        authors.append(name)
        for author in author_name_pattern.findall(paper_raw):
            authors.append(author)
        title = title_pattern.search(paper_raw).group(1)

        # find publish info & year, try two patterns
        try:
            volume, year = publish_info_pattern_A.search(paper_raw).group
            publishInfo = f"{volume} ({year})"
        except:
            try:
                volume, year, pagination = publish_info_pattern_B.search(
                publishInfo = f"{volume} {year}: {pagination}"
            except:
                volume, year = publish_info_pattern_C.search(paper_raw).g
                publishInfo = f"{volume} {year}"
        year = int(year)
        if year < 2020:
```

```
            continue

        papers.append({
            "authors": authors,
            "title": title,
            "publishInfo": publishInfo,
            "year": year,
        })

    return {
        "name": name,
        "orcid": orcid,
        "papers": papers
    }

# pprint(handle_url(author_urls[0]))
```

去重，拉取所有作者

In [ ]:
```
cleaned_author_urls = list(set(author_urls))
print(len(cleaned_author_urls))
authors = []
for url in cleaned_author_urls:
    authors.append(handle_url(url))
```

19

依照要求保存文件

In [ ]:
```
# save to reserachers.json
with open("researchers.json", "w") as f:
    f.write(json.dumps(authors, indent=2))
```

尾声：不使用 DOM parser 而暴力使用 Regex, 虽说是不需要依赖第三方库（可以现场学习），但无疑极大增加编程中的阻力，而且并没有什么学习的价值。