1. 实验目的

给定评课社区网站 ，需要设计一个网站遍历策略，爬取至少 200 个课程的详细信息，记录于 json 格式的文件中，需要提取的信息入下图红框所示：

2. 实验思路

2.1. 整体结构

crawler.py 为爬虫源代码

pksq.json 为存放课程信息的文件

2.2. module

```
4   from bs4 import BeautifulSoup
5   import requests
6   import json
7   from multiprocessing.dummy import Pool
8   import time
9
```

(1) requests：代理发送请求，获得响应文

(2) BeautifulSoup：解析页面源代码，用以提取想要的 html 对象，如 url、text 等

(3) json：将字典对象转换为 json 格式、

(4) time：计算程序运行时间

2.3. function

```
def crawl(url):
def get_url(url):
if __name__ == '__main__':
```

(1) crawl：爬取课程主页的文本

(2) get_url：爬取主网页的所有课程主页链接并调用 crawl 函数爬取课程信息并返回本次爬取的课程信息组成的列表

(3) main：主函数，用于翻页直至课程数大于 200

2.4. 具体思路

主函数从 page=1 开始

(1) 首先，调用函数 get_url 获取该 page 的所有课程页面的 url 链接并返回爬取的课程信息组成的列表，并将其与主函数的总结果列表 lec_list 相加，同时在开头和结尾调用 time 库计时并输出

(2) 然后，在函数 get_url 中调用函数 crawl 返回课程信息组成的对象，将其加入列表

(3) 计算总结果列表 lec_list 的长度，如果大于 200，转为 json 格式并保存到目标文件，程序结束，否则 page+=1，转(1)

3. 缺点与改进

原本为单线程，速度过慢，改为多线程爬虫（在函数 get_url 中实现多线程调用 crawl 函数），设定线程数为 10

```
# 定义10个线程池
pool = Pool(10)
# 利用map让线程池中的所有线程‘同时’执行crawl函数
pool.map(crawl, url_list)
```

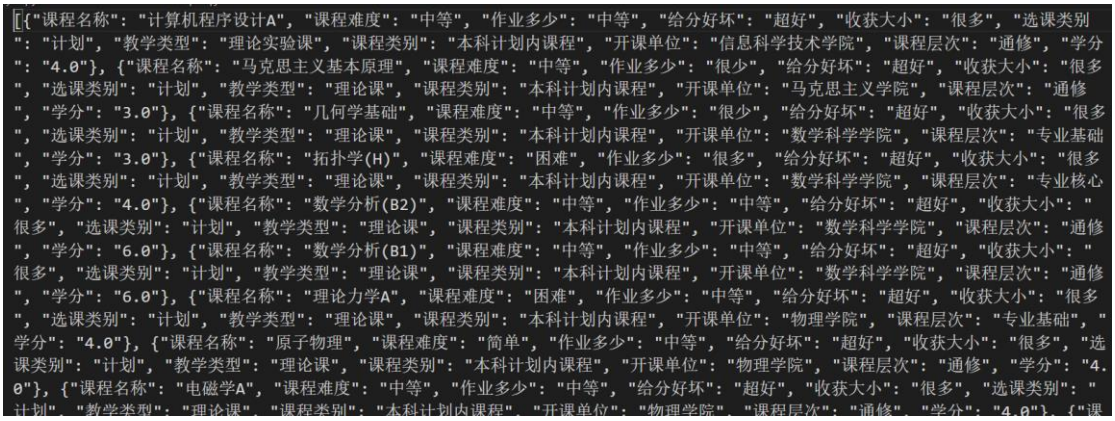
所用时间(单位 s)：

```
111.3022530078888
Press any key to continue . . .
```

是单线程时间的$\frac{1}{8}$左右，比$\frac{1}{10}$大的原因推测为 python 线程转换需要一定时间，总体上时间效率改进显著，但是有时会遇到网站反爬措施限制，此时需减少线程数或者设置爬一个网页一次后停顿时间

4. 实验结果

详见 pksq.json 文件，以下为部分截图



5. 参考资料

[1] http://t.csdn.cn/x1jYI

[2] http://t.csdn.cn/g0q1f

[3] http://t.csdn.cn/kGZEv