Hw#2, NLP@CGU Spring 2023 LINK: paste your link here

https://colab.research.google.com/drive/1fN-LoT3F7N6Yv0Oy7v4nTfusl5E30ys6?usp=sharing

Student ID: B0928001 **Name**: 賴霆瑞

```
import requests
from bs4 import BeautifulSoup
import networkx as nx

# specify the base URL and the range of pages you want to scrape
base_url = "https://movies.yahoo.com.tw/movieinfo_main/"
start_page = 1
end_page = 10

# define a Graph object
G = nx.Graph()

# loop through each page and scrape the movie information
for page in range(start_page, end_page+1):
    url = base_url + str(page)
    response = requests.get(url)
    soup = BeautifulSoup(response.content, "html.parser")
    if (soup.find("div", class_="movie_intro_info") == None):
        continue
    # add the URL to the Graph
    G.add_node(url)
    # find the movie information on the page
    doc_id = page
    cname = soup.find("div", class_="movie_intro_info").find("h1").text.strip()
    ename = soup.find("div", class_="movie_intro_info_r").find("h3").text.strip()
    label = soup.find("div", class_="level_name").text.strip()
    intro = soup.find("span", id="story").text.strip().replace('\n\n', '')
    released_date = soup.find("div", class_="movie_intro_info_r").find_all("span")[0].text.strip().replace('上映日期：', '')

    # get all the links on the page and add them to the Graph
    links = []
    for link in soup.find_all("a"):
        href = link.get("href")
        text = link.text.strip()
        if href and text:
            links.append({"href": href, "text": text})
            G.add_edge(url, href)

    # print the movie information with the calculated PageRank
    pagerank = nx.pagerank(G)
    print({
        "doc_id": doc_id,
        "cname": cname,
        "ename": ename,
        "pagerank": pagerank[url], # use the URL as the key to get the PageRank
        "label": label,
        "intro": intro,
        "released_date": released_date,
        "links": links
    })
```

```
{'doc_id': 1, 'cname': '一世狂野', 'ename': 'Blow', 'pagerank': 0.46051496928893304, 'label': '劇情', 'intro': '喬治戎格一生都在追求所謂的美國夢
{'doc_id': 2, 'cname': '玩命關頭', 'ename': 'The Fast and the Furious', 'pagerank': 0.2410828215489779, 'label': '動作', 'intro': '唐米尼杜洛托
{'doc_id': 3, 'cname': '戰雲密佈', 'ename': 'Storm Catcher', 'pagerank': 0.14111163386770906, 'label': '動作', 'intro': '美國空軍最高機密的隱形
{'doc_id': 4, 'cname': '騎士風雲錄', 'ename': "A Knight's Tale", 'pagerank': 0.11540232994260224, 'label': '動作', 'intro': '14世紀中古時期的社
{'doc_id': 5, 'cname': '金法尤物', 'ename': 'Legally Blonde', 'pagerank': 0.09584932468224716, 'label': '喜劇', 'intro': '在【歡樂谷】、【危險性
{'doc_id': 6, 'cname': '瘋狂世界', 'ename': 'Rat Race', 'pagerank': 0.07773888216344625, 'label': '冒險', 'intro': '薇拉貝克(琥碧戈珀飾）來到賭城
{'doc_id': 7, 'cname': '震撼教育', 'ename': 'Training Day', 'pagerank': 0.07119912160242191, 'label': '劇情', 'intro': '在【震撼教育】中，丹佐扮
{'doc_id': 8, 'cname': '神鬼第六感', 'ename': 'The Others', 'pagerank': 0.055190541423527265, 'label': '劇情', 'intro': '一個名叫葛瑞絲(妮可基嫚
{'doc_id': 10, 'cname': '觸不到的戀人', 'ename': 'il mare', 'pagerank': 0.04979482266247514, 'label': '奇幻', 'intro': '1997年剛搬家的星賢在信箱
```

爬取Yahoo!電影資料

```
import requests
from bs4 import BeautifulSoup
import networkx as nx
import json

# specify the base URL and the range of pages you want to scrape
base_url = "https://movies.yahoo.com.tw/movieinfo_main/"
start_page = 1
end_page = 15067
```

```
# define a Graph object
G = nx.Graph()

# loop through each page and scrape the movie information
for page in range(start_page, end_page+1):
        url = base_url + str(page)
        response = requests.get(url)
        soup = BeautifulSoup(response.content, "html.parser")
        if (soup.find("div", class_="movie_intro_info") == None):
                continue
        # add the URL to the Graph
        G.add_node(url)
        # find the movie information on the page
        doc_id = page
        cname = soup.find("div", class_="movie_intro_info").find("h1").text.strip()
        ename = soup.find("div", class_="movie_intro_info_r").find("h3").text.strip()
        label = soup.find("div", class_="level_name").text.strip()
        intro = soup.find("span", id="story").text.strip().replace('\n\n', '')
        released_date = soup.find("div", class_="movie_intro_info_r").find_all("span")[0].text.strip().replace('上映日期：', '')

        # get all the links on the page and add them to the Graph
        links = []
        for link in soup.find_all("a"):
                href = link.get("href")
                text = link.text.strip()
                if href and text:
                        links.append({"href": href, "text": text})
                        G.add_edge(url, href)

        # calculate the PageRank for the page
        pagerank = nx.pagerank(G)

        # create a dictionary for the movie information
        movie_info = {
                "doc_id": doc_id,
                "cname": cname,
                "ename": ename,
                "pagerank": pagerank[url], # use the URL as the key to get the PageRank
                "label": label,
                "intro": intro,
                "released_date": released_date,
                "links": [link["href"] for link in links] # store only the hrefs of the links
        }

        # append the movie information to the existing JSON file
        with open("movies.json", "a", encoding="utf-8") as f:
                json.dump(movie_info, f, ensure_ascii=False)
                f.write("\n") # add a new line at the end of each JSON object to separate them



### 中文分詞後，建立 Inverted Index
import json
import jieba
from collections import defaultdict

# load the movie data from the JSON file
with open('movies.json', 'r', encoding='utf-8') as f:
        movies = [json.loads(line) for line in f]

# define a tokenizer function using jieba
def tokenize(text):
        return list(jieba.cut(text))

# build the inverted index
inverted_index = defaultdict(list)
for movie in movies:
        # tokenize the movie intro, cname, ename, and label fields
        tokens = tokenize(movie['intro'] + movie['cname'] + movie['ename'] + movie['label'])
        # add each token to the inverted index along with the movie ID
        for token in tokens:
                inverted_index[token].append(movie['doc_id'])

# save the inverted index to a JSON file
with open('inverted_index.json', 'w', encoding='utf-8') as f:
        json.dump(inverted_index, f, ensure_ascii=False)
        f.write("\n") # add a new line at the end of each JSON object to separate them
```

```
Building prefix dict from the default dictionary ...
DEBUG:jieba:Building prefix dict from the default dictionary ...
Dumping model to file cache /tmp/jieba.cache
DEBUG:jieba:Dumping model to file cache /tmp/jieba.cache
Loading model cost 1.005 seconds.
```

```
DEBUG:jieba:Loading model cost 1.005 seconds.
Prefix dict has been built successfully.
DEBUG:jieba:Prefix dict has been built successfully.
```

```
### 利用 PageRank 演算法來排序
import json

# read in the movie information from the JSON file
movies = []
with open("movies.json", "r", encoding="utf-8") as f:
    for line in f:
        movie_info = json.loads(line)
        movies.append(movie_info)

# sort the movies based on their PageRank scores
movies_sorted = sorted(movies, key=lambda x: x["pagerank"], reverse=True)

# print out the top 10 movies
for i, movie in enumerate(movies_sorted[:50]):
    print(f"{i+1}. {movie['cname']} ({movie['released_date']}) - PageRank score: {movie['pagerank']:.4f}")
```

```
1. 黑狗來了 (2004-03-27) - PageRank score: 0.4606
2. 我和吸血鬼有份合約 (2001-11-24) - PageRank score: 0.4606
3. 北京樂與路 (2001-11-10) - PageRank score: 0.4605
4. 幸福騙局 (2018-11-30) - PageRank score: 0.4605
5. 麻雀變公主 (2001-09-01) - PageRank score: 0.4605
6. 一世狂野 (2001-10-12) - PageRank score: 0.4605
7. 鬼訊號 (2005-01-14) - PageRank score: 0.4605
8. 鬼仔 (2013-11-29) - PageRank score: 0.4605
9. 英倫對決 (2017-10-13) - PageRank score: 0.4605
10. 野鴨變鳳凰：冰上逆襲
                                        (2021) ((2021)) - PageRank score: 0.4605
11. 泰德拉索：錯棚教練趣事多
                                        (2021) ((2021)) - PageRank score: 0.4603
12. 野鴨變鳳凰：冰上逆襲
                                        (2022) ((2022)) - PageRank score: 0.2485
13. 實尾島風雲 (2004-04-02) - PageRank score: 0.2422
14. 玩命關頭 (2001-10-13) - PageRank score: 0.2411
15. 關鍵琴聲 (2014-01-17) - PageRank score: 0.2394
16. 千禧曼波 (2001-11-17) - PageRank score: 0.2363
17. BJ單身日記(2001) (2001-10-27) - PageRank score: 0.2343
18. 泰德拉索：錯棚教練趣事多
                                        (2020) ((2020)) - PageRank score: 0.2319
19. 血肉森林 (2004-12-24) - PageRank score: 0.2288
20. 這旅程使命必達 (2018-12-07) - PageRank score: 0.2288
21. 徐自強的練習題 (2017-08-11) - PageRank score: 0.2216
22. 晚孃 (2001-11-03) - PageRank score: 0.2209
23. 屏息 (2013-12-13) - PageRank score: 0.1764
24. 變身特務 (2020-01-22) - PageRank score: 0.1717
25. 見鬼2 (2004-03-26) - PageRank score: 0.1602
26. 青禾男高 (2017-07-21) - PageRank score: 0.1597
27. 救世主 (2001-12-07) - PageRank score: 0.1570
28. 人間有情天 (2001-11-17) - PageRank score: 0.1530
29. 貝蒂費雪的世界 (2004-12-10) - PageRank score: 0.1516
30. 追夢高手 (2001-11-30) - PageRank score: 0.1447
31. 心碎效應 (2022-09-23) - PageRank score: 0.1441
32. 戰雲密佈 (2001-10-13) - PageRank score: 0.1411
33. 御史與祚怡
                                        (2021) ((2021)) - PageRank score: 0.1395
34. 功夫 (2004-12-24) - PageRank score: 0.1244
35. 老師君主 (2018-11-16) - PageRank score: 0.1228
36. 美國派2 (2001-11-10) - PageRank score: 0.1211
37. 行動代號：狼狩獵 (2022-09-30) - PageRank score: 0.1204
38. 騎士風雲錄 (2001-10-19) - PageRank score: 0.1154
39. 人性污點 (2004-04-02) - PageRank score: 0.1136
40. 如蝶翩翩
                                        (2021) ((2021)) - PageRank score: 0.1118
41. 鬼計神偷 (2001-10-06) - PageRank score: 0.1098
42. 愛貓之城 (2017-08-04) - PageRank score: 0.1080
43. 屍房宴 (2014-01-17) - PageRank score: 0.1072
44. 大發不動產
                                        (2021) ((2021)) - PageRank score: 0.1043
45. 哥兒們(2001) (2001-11-17) - PageRank score: 0.1042
46. 靈魂的重量 (2004-04-16) - PageRank score: 0.0989
47. 艾蜜莉的異想世界 (2002-03-08) - PageRank score: 0.0989
48. 金法尤物 (2001-10-19) - PageRank score: 0.0958
49. 少林足球 (2001-08-24) - PageRank score: 0.0944
50. 童話・世界 (2022-10-07) - PageRank score: 0.0930
```

```
### 定義搜尋Function
import json

def search_movies(query):
    # load the movie info from the JSON file
    with open("movies.json", "r", encoding="utf-8") as f:
```

```
        movies = [json.loads(line) for line in f]

    # create a dictionary of {doc_id: PageRank value} pairs
    pageranks = {movie["doc_id"]: movie["pagerank"] for movie in movies}

    # create a list of movies that match the search query
    matches = [movie for movie in movies if any([query.lower() in str(value).lower() for key, value in movie.items()])]

    # sort the matches by their PageRank values
    sorted_matches = sorted(matches, key=lambda x: pageranks.get(x["doc_id"], 0), reverse=True)

    # print the search results
    print("您的搜尋結果 (Sorting by PageRank Value)：")
    print(f"共 {len(sorted_matches)} 筆，符合'{query}'  - - - 共 indexing {len(movies)} 筆電影資料")
    for movie in sorted_matches:
        intro = movie.get("intro", "")
        print(f"{movie['doc_id']} ({pageranks.get(movie['doc_id'], 0)}): {movie['cname']} ({movie['ename']}) - {intro}")


### 輸入搜尋關鍵字，任何字都可以
search_movies("劇情")
```

```
您的搜尋結果 (Sorting by PageRank Value)：
共 5390 筆，符合'劇情'  - - - 共 indexing 12240 筆電影資料
515 (0.4606015592685615): 黑狗來了 (Comes The Black Dog) - 小建築承包商永吉（太保飾），嗜賭如命，因無力償還賭債，偷走欠錢的收據而被黑道老大黑
15 (0.4605565324791542): 我和吸血鬼有份合約 (Shadow of the Vampire) - 1921年默片時期，德國最重要的導演F. W.穆瑙，決定不顧【德古拉】Dracula原作
9002 (0.46054231349302666): 幸福騙局 (Euforia) - ★2018坎城影展一種注目單元入圍，通俗不凡的家庭悲喜劇
★《安娜的美麗人生》《雨人》演而優則導的威尼斯影后薇拉莉葛琳諾第二部動人長片作品
★聞天祥老師推薦「2018坎城影展9部可能被忽略的佳作」之一！
★《愛情流彈》義大利男神 里卡多史卡馬西奧+《願望咖啡館》”魔鬼”影帝 瓦里歐馬斯坦德，收放自如演技搭檔演出兄弟，被譽為「完全掌控觀眾的笑聲和淚水
★動人家庭劇宛如《愛情流彈》般優雅風趣，絕美場景亦同《以你的名字呼喚我》般田園詩意幸福，是一陣狂喜過後，徒留悲傷的騙局…
馬德奧是一名的年輕企業家，思想開放、迷人又有活力，私下則是從成功和性愛的追求中尋找快感的男同志。而他的哥哥艾托雷在他們故鄉小鎮上的中學教書，因
1 (0.46051496928893304): 一世狂野 (Blow) - 喬治戎格一生都在追求所謂的美國夢，也就是享受美好富裕的生活，但是他卻不願像他父親那樣一輩子都只是個
14001 (0.4604764848535408): 野鴨變鳳凰：冰上逆襲
                                                (2021) (The Mighty Ducks: Game Changers) - 《野鴨變鳳凰：冰上逆襲》講述現今青少年運動世界的故事。

在明尼蘇達州，野鴨隊目前已經蛻變成一隊競爭力極強，實力雄厚的少年冰上曲棍球球隊。12歲的艾文莫羅自小在野鴨隊參與訓練，被踢出球隊後傷心不已。他的
14002 (0.24850175651987502): 野鴨變鳳凰：冰上逆襲
                                                (2022) (The Mighty Ducks: Game Changers) - 《野鴨變鳳凰：冰上逆襲》講述現今青少年運動世界的故事。
18 (0.23625393700326205): 千禧曼波 (Millennium Mambo) - 年輕生命的消耗像是一首風馳電掣的搖滾樂，以熱情洋溢的節奏，瘋狂起始，然後又在某個渾然忘
9007 (0.2287719532585709): 這旅程使命必達 (The Last Suit) - ★亞特蘭大影展觀眾票選獎
★耶路撒冷影展最佳劇情片
★釜山影展入選觀摩
★年度最溫馨感人的西班牙語電影
★二戰後最療癒的奇幻旅行八十八歲的猶太裔阿根廷裁縫師亞伯拉罕，離開了阿根廷，決心啟程前往波蘭的羅茲市尋找七十多年前救他一命的朋友。從阿根廷抵達
12 (0.22092020318475641): 晚孃 (Jan Dara) - "真"一出生，他的母親"Dara"便與世長辭。有人說，他的一生從此受到詛咒。由這時候開始，"Dara"的丈夫老爺
不過，老爺對"真"的仇恨，跟他在「Visanan莊園」內的荒淫無度卻毫無關係。"真"最早的童年回憶，是目睹老爺在他亡妻的肖像前跟女佣打得火熱。事實上，老
5002 (0.1764479511846573): 屏息 (No Breathing) - ★南韓影史首部游泳競技電影，連續登上網路搜尋榜NO.1！
★打敗《地心引力》等好萊塢強片，數度蟬聯預售排行冠軍！
★南韓上映首週，票房吸金19億，榮登本土新片票房冠軍！
★「大勢歌手」徐仁國 V.S.「少女時代」俞利首度主演電影！
★李鍾碩、徐仁國等多位極品美男看到飽，放送養眼胸腹肌！
★預告發佈首日，點閱率爆破87萬次，刷新《偉大的隱藏者》歷史紀錄！還沒結束！我們的比賽，現在正要開始！元日（徐仁國飾演）曾經是前途光明的游泳冠軍
《屏息》是南韓影史首部以「游泳競技」為主題的電影，也是2013南韓下半年最受觀眾期待的電影之一。由新銳導演趙容善執導，劇本則由劉英亞執筆。這位曾打
13 (0.15301686110820453): 人間有情天 (The Son's Room) - (2001年坎城影展最佳影片金棕櫚獎) 心理醫師喬凡尼一家四口，居住在義大利北部美麗的濱海城
1004 (0.1516240170734289): 貝蒂費雪的世界 (Betty Fisher) - 貝蒂是法國知名的小說家，獨立扶養四歲兒子約瑟夫，過著幸福快樂的日子；隔著幾條街外的
220 (0.14472077124104016): 追夢高手 (Hardball) - 一個沉迷頗深的賭徒康納歐尼爾（基努李維飾），他非常喜歡在運動比賽上下賭注，弄得他常常生活拮据
518 (0.11356556573673988): 人性污點 (The Human Stain) - 美國雅典娜學院猶太裔教務長柯曼席爾克年過六十，在學校教授古典文學，課堂上對於時常缺席沒
12004 (0.11180739728952467): 如蝶翩翩
                                                (2021) (Navillera) - 這部暖心韓劇由《隧道》編劇李恩美主筆，宋江、朴仁煥、羅文姬、洪承希主演。改
22 (0.10984932310397796): 鬼計神偷 (The Score) - 尼克魏斯準備洗手不幹，他打算和他女友黛安退隱江湖，安安份份地在蒙特婁經營一家爵士餐廳，但是他
12005 (0.10428108764995832): 大發不動產
                                                (2021) (Sell Your Haunted House) - 童顏女神「張娜拉」和偶像男神「鄭容和」攜手打造的驅魔喜劇！劇
225 (0.10417304793010666): 哥兒們(2001) (So-Called Friends) - 台灣青年小林(林強飾）跟著電影公司外景隊赴大陸拍片，擔任攝影助理的工作，因為私自

小林醫院裡認識了在少年宮當老師的女子李佳(麗鳳飾），雙方都對對方有好感，開始了短暫的交往。小林與小周一塊在北京搞丟了朋友的吉普車，小周只好賣掉

在北京的日子裡有快樂也有悲傷，小林的證件終於發下來了，面對短暫相處卻即將分離的李佳，兩人都有感傷，卻無法對未來有任何的期許。最後小林告別了北京

在北京胡同內來自兩岸的年輕人偶爾相遇，單純的上演了一段純屬「哥兒們」之間的生活週記，各自述說著自身的快樂與苦悶，對他們來說未來似乎近在咫尺，又
519 (0.09894242464841585): 靈魂的重量 (21 Grams) - 本片由【愛情像母狗】編劇吉勒莫亞瑞格所編寫，故事敘述西恩潘、班尼西歐岱托羅和納歐密華茲三個
228 (0.0988572609113828): 艾蜜莉的異想世界 (Amelie) - 尋找你所不知道的東西，是困難的…艾蜜莉住在巴黎，有個屬於她自己的小天地裡，她在咖啡館上班
14005 (0.09299070919809506): 童話・世界 (Fantasy・World) - 初出社會的人權律師張正煦（張孝全 飾），首次承辦案件便是震驚社會一時的補教名師性侵案

【關於電影】

一部以律師、訴訟、誘姦未成年少女議題的國片《童話・世界》，成為今年台北電影節閉幕片，同時入圍最佳影片、最佳男主角、最佳女主角、最佳女配角、最佳

以律師經歷出發的《童話・世界》劇本　網羅國片影帝后級演員加持
```

```
### 輸出搜尋結果呈現
import json
import re

def search_movies(query):
```

```
        # load the movie info from the JSON file
        with open("movies.json", "r", encoding="utf-8") as f:
            movies = [json.loads(line) for line in f]

        # create a dictionary of {doc_id: PageRank value} pairs
        pageranks = {movie["doc_id"]: movie["pagerank"] for movie in movies}

        # create a list of movies that match the search query
        matches = [movie for movie in movies if any([query.lower() in str(value).lower() for key, value in movie.items()])]

        # sort the matches by their PageRank values
        sorted_matches = sorted(matches, key=lambda x: pageranks.get(x["doc_id"], 0), reverse=True)

        # calculate precision and recall
        relevant_docs = set([movie['doc_id'] for movie in movies if query.lower() in str(movie).lower()])
        retrieved_docs = set([movie['doc_id'] for movie in matches])
        true_positives = relevant_docs.intersection(retrieved_docs)
        precision = len(true_positives) / len(matches) if len(matches) > 0 else 0
        recall = len(true_positives) / len(relevant_docs) if len(relevant_docs) > 0 else 0

        # print the search results and evaluation metrics
        print("您的搜尋結果 (Sorting by PageRank Value)：")
        print(f"共 {len(sorted_matches)} 筆，符合'{query}'  - - - 共 indexing {len(movies)} 筆電影資料")
        for movie in sorted_matches:
            intro = movie.get("intro", "")
            print(f"{movie['doc_id']} ({pageranks.get(movie['doc_id'], 0)}): {movie['cname']} ({movie['ename']}) - {intro}")

        print(f"\nPrecision: {precision:.0%}")
        print(f"Recall: {recall:.0%}")


    ### 輸入搜尋關鍵字，任何字都可以
    search_movies("Blow")
```

```
★瑞秋懷茲、莫妮卡貝露琪，兩大女生聊手機感。艾蜜絲凱薩琳，被上級指派到波西尼亞，加入聯合國維安部隊，協助里建的監督工作。她的任務是訓練波西尼亞
4766 (0.00017984344657567337): 戀戀海灣 (As the Winds Blow) - 在海邊出生，幼時隨著母親北上定居，米雪早已忘記自己體內的海洋基因。在母親的命令下

Precision: 100%
```

7