Assignment 3

Submission Instructions

- Keep your script in one *.py file. The entire program must run from start to finish without error.
- Upload your script to the learning hub drop box for Assignment 3.

Note: You are encouraged to work with others but DO NOT SHARE COPIES of YOUR WORK ELECTONICALLY OR IN ANY OTHER FORM.

Please start early and work consistently on this homework each day. I recommend completing at least one section a day as a minimum.

Extra Requirement:

Do not scrape a library site. Do not scrape any sites that we have scraped in class.

Part A

Write an application that scrapes uniform rows of content from three pages of a site that you are interested in. Do not use a site that has been presented in class. Please find a site that will allow you to fulfill these requirements. The site you scrape cannot be one that was used in the class. Your application must:

a) Search for content on the site. This will involve automation to send a search term to a search box and clicking a search button. **(10 marks)**

b) Scrape 3 pages of the site. To do this, your application automates the clicking of a button or link to advance to the next page to scrape the second and third pages. **(10 marks)**

c) During each iteration for every row of content scraped, store the parsed and cleansed content in a list. **(10 marks)**

d) Build a dataframe from your list **(5 marks).**

e) Save your data frame into a CSV file. Then read your content into a new data frame and display the first two rows and last two rows of the new data frame. **(5 marks)**

Part B

h) On a different site, find and scrape some content that you can summarize dynamically and present in a graph. Be sure to use an aggregation (summary) of some kind. Keep it simple but sensible. **(25 marks)**

To Avoid Deductions

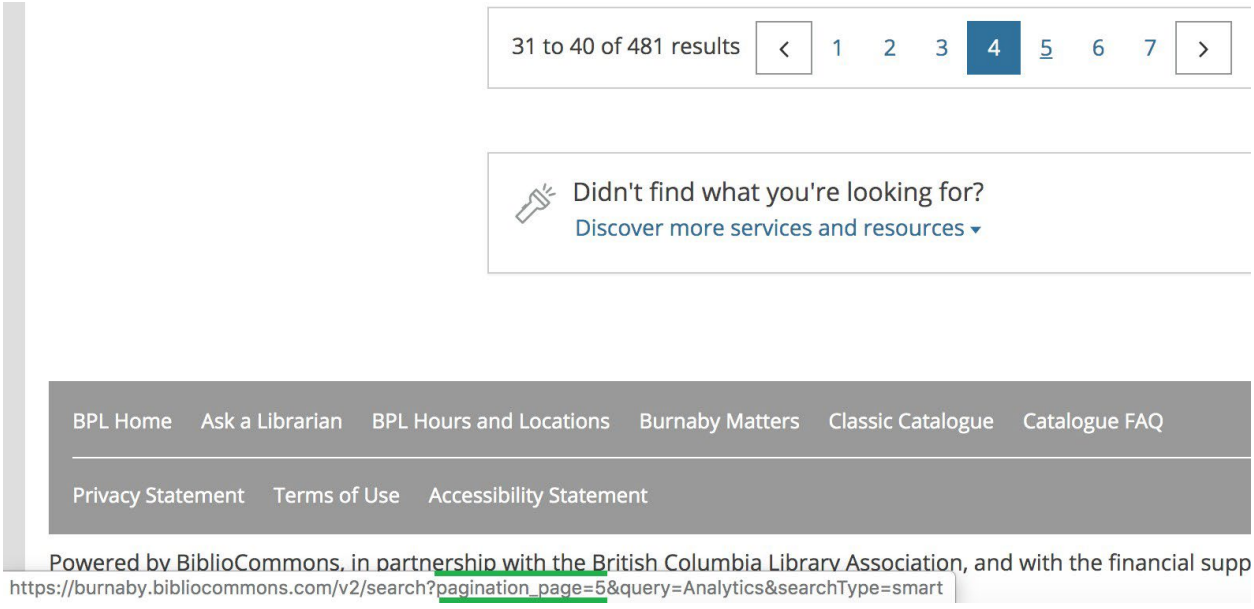
- Use proper descriptive naming for variables, functions and classes.
- Eliminate duplicate code.
- Use functions to encapsulate and organize your code where possible.

Hint 1: Adding a Row to a DataFrame

To add a row to a DataFrame examine example 11 day 2.

Hint 2: Advancing to the Next Page

If you are having trouble advancing to a next page using a hyperlink you can navigate to the page directly. For example, when you want to go to page 5 from page 4 you can manipulate the parameter in the hyperlink that sets the page number. This parameter will be different on every web page that uses hyperlinks so to view it, hover over the link with your cursor and inspect the hyperlink at the bottom left of the browser window.



Now you can build the hyperlink in a manner similar to the following:

```
pageNum +=1
URL_NEXT = "https://burnaby.bibliocommons.com/v2/search?query="\
      + SEARCH_TERM + "&searchType=smart&pagination_page="
URL_NEXT = URL_NEXT + str(pageNum)
browser.get(URL_NEXT)
```

Here is the full example:

```
import time
from selenium import webdriver
from bs4 import BeautifulSoup
import re

# driver = webdriver.Chrome(ChromeDriverManager().install())
PATH = "C:\\datasets\\chromedriver.exe"
URL = "https://www.bpl.bc.ca/events/"
browser = webdriver.Chrome(PATH)
browser.get(URL)

# Give the browser time to load all content.
time.sleep(1)

SEARCH_TERM = "Analytics"
search = browser.find_element_by_css_selector("input")

search.send_keys(SEARCH_TERM)

# Find the search button - this is only enabled when a search query is entered
button = browser.find_element_by_css_selector("button")
button.click()  # Click the button.
time.sleep(3)

# content = browser.find_elements_by_css_selector(".cp-search-result-item-content")

pageNum = 1;
for i in range(0, 3):
    content = browser.find_elements_by_css_selector(".cp-search-result-item-content")

    for e in content:
        textContent = e.get_attribute('innerHTML')

        # Beautiful soup removes HTML tags from our content if it exists.
        soup = BeautifulSoup(textContent, features="lxml")
        rawString = soup.get_text().strip()

        # Remove hidden characters for tabs and new lines.
        rawString = re.sub(r"[\n\t]*", "", rawString)

        # Replace two or more consecutive empty spaces with '*'
        rawString = re.sub('[ ]{2,}', '*', rawString)
        print(rawString)
        print("***")
```

```
    pageNum += 1
    URL_NEXT = "https://burnaby.bibliocommons.com/v2/search?query=" \
            + SEARCH_TERM + "&searchType=smart&pagination_page="
    URL_NEXT = URL_NEXT + str(pageNum)
    browser.get(URL_NEXT)

    print("Count: ", str(i))
    time.sleep(3)

print("done loop")
```

New Code

Note: If you see the DeprecationWarnings above, it means that you are using the new edition of selenium. You can use the following code:

```
import time
from selenium import webdriver
from bs4 import BeautifulSoup
import re
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
# driver = webdriver.Chrome(ChromeDriverManager().install())
Path = "/Python/ChromeDriver/"
File = 'chromedriver.exe'
URL = "https://www.bpl.bc.ca/events/"
browser = webdriver.Chrome(service=Service(Path+File))
browser.get(URL)

# Give the browser time to load all content.
time.sleep(1)

SEARCH_TERM = "Analytics"
search = browser.find_element(By.CSS_SELECTOR,"input")

search.send_keys(SEARCH_TERM)

# Find the search button - this is only enabled when a search query is
entered
button = browser.find_element(By.CSS_SELECTOR,"button")
button.click()  # Click the button.
time.sleep(3)

# content = browser.find_elements_by_css_selector(".cp-search-result-item-
content")

pageNum = 1;
for i in range(0, 3):
    content = browser.find_elements(By.CSS_SELECTOR,".cp-search-result-item-
content")
```

```
    for e in content:
        textContent = e.get_attribute('innerHTML')

        # Beautiful soup removes HTML tags from our content if it exists.
        soup = BeautifulSoup(textContent, features="lxml")
        rawString = soup.get_text().strip()

        # Remove hidden characters for tabs and new lines.
        rawString = re.sub(r"[\n\t]*", "", rawString)

        # Replace two or more consecutive empty spaces with '*'
        rawString = re.sub('[ ]{2,}', '*', rawString)
        print(rawString)
        print("***")

    pageNum += 1
    URL_NEXT = "https://burnaby.bibliocommons.com/v2/search?query=" \
               + SEARCH_TERM + "&searchType=smart&pagination_page="
    URL_NEXT = URL_NEXT + str(pageNum)
    browser.get(URL_NEXT)

    print("Count: ", str(i))
    time.sleep(3)

print("done loop")
```