Advanced Web Development and Web Scraping
Spring 2020
Assignment #7 – Web Scraping Assignment

Note: For all assignments, *get* the web page using python's request library, and include an appropriate header in the request.

1. Scrape the sample Schedule page (https://gdancik.github.io/CSC-301/data/notes/schedule.html) to output the number of courses being taught and the *total* number of credits, in the following format:

   ```
   Dr. Dancik is teaching 3 courses (9 credits)
   ```

   Note: Your scraper should work for different data (e.g., a different instructor, or a different number of courses. However, the page will always have the same format (e.g., There will be a single table with the list of courses, with columns for Course, Time, and # Credits, in that order).

   Note: In order to add the credits, you will need to convert each 'number' to an integer, using the *int* function, e.g., `int('3')` will return 3.

2. Scrape the title and rating for 5 movies from IMDB, whose links are given below, and construct a bar graph that shows the rating for each movie. Give your graph an informative title. The following links should be used:

   - https://www.imdb.com/title/tt0109830/?ref_=fn_al_tt_1
   - https://www.imdb.com/title/tt0076759/?ref_=fn_tt_tt_1
   - https://www.imdb.com/title/tt0368226/?ref_=nv_sr_2
   - Select another movie from IMDB and include the URL
   - Select another movie from IMDB and include the URL

   In order to do this, you should create a list containing the URLs and iterate through the list to scrape the relevant information for each movie. Note that all pages have the same format. **After submitting a request to a page, you must sleep for 1 second so that you do not overburden IMDB's servers.** This is done by importing the *time* module and calling *time.sleep()*:

   ```
   import time
   time.sleep(1)
   ```

   Notes: (1) The rating will need to be converted to a float (decimal) using the float function (e.g., `float("3.1")` will return the number 3.1); (2) The title strings may

contain a '\xa0', which is code for a non-breaking space. These do not have to be removed, but if you want to remove them, you can use python's *strip* method.

3. Scrape the job listing site *Indeed* (https://www.indeed.com/) to find jobs for a search of your choice. You will need to copy the URL for your search and use it in your Python script. Note that because *Indeed* uses the GET method to retrieve information, user input is visible in the URL. For example, a search for "computer programmer" in "Hartford, CT" has the following URL: https://www.indeed.com/jobs?q=computer+programmer&l=Hartford%2C+CT

   Your web scraper should create a *pandas* data frame that contains the following information.

	Title	Company	Location	Salary
0	CAD/CAM Programmer	INDUSTRIAL HEATER CORP- HI TECH FABRICATING INC	Cheshire, CT 06410	$16 - $17 an hour
1	Entry Level Computer Programmer	Revature	Hartford, CT	?
2	Fire Alarm Inspector	Fire Protection Testing	Cheshire, CT 06410	$40,000 - $50,000 a year

   You therefore will need to extract the job title, company, location, and salary from each job listing. Note that you can use the `get_text()` method to get the innerText of an element (which may span multiple elements). For example, `soup.div.get_text()` will return a string containing all of the text in the first div. The use of `get_text()` is recommended, because it works in all cases (it works for elements that do and do not have children). Not all job listings have salary information; if they do not, you should display a question mark (?)

   Your output should be stripped of extra white space.

   Note: if using Jupyter Notebook, a dollar sign ($) in a pandas data frame is interpreted as denoting a mathematical expression, which effects the formatting. In order to turn this formatting off, run the following statement after importing the pandas module:

```
pd.options.display.html.use_mathjax = False
```