C996: Project Write Up

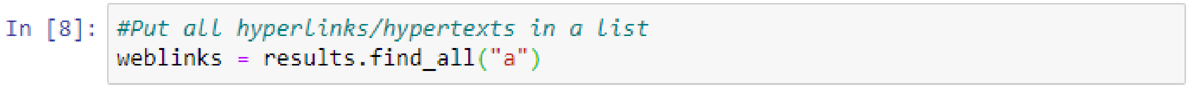
James Shea

Task A:

The Python program extracts the web links from the HTML code of the “Current Estimates” first using urllib request and then utilizing Beautiful Soup to help read the HTML from it. I used the Beautiful Soup documentation available at https://www.crummy.com/software/BeautifulSoup/bs4/doc/. Then, find all is used to search for “a” tag which allows me to collect all hyperlinks inside.

Task B:

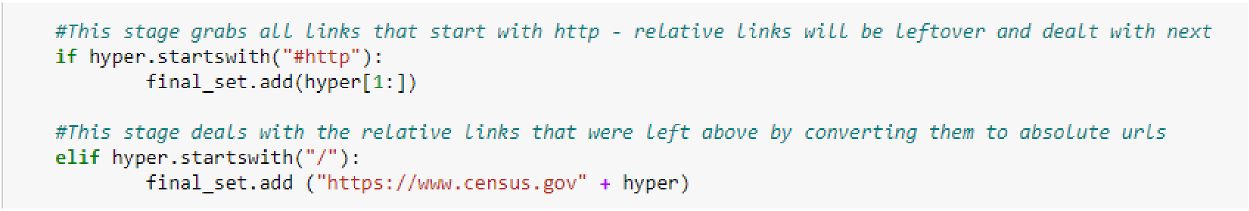
The code used the criteria of the html tag “a” which indicates hyperlinks. The code to do so is screenshot below:



```
In [8]: #Put all hyperlinks/hypertexts in a list
        weblinks = results.find_all("a")
```

Task C:

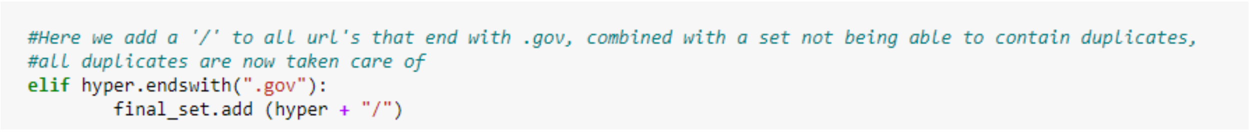
The loop shown below first deals with all links starting with http. Then, the relative links are converted to absolute URLs . This works as relative URLs are not able to begin with HTML according to coffeecup.com, and all relative links are relative to https://www.census.gov. Relative links are saved as absolute URIs in the output file by using the following code in the screenshot below:



```
#This stage grabs all links that start with http - relative links will be leftover and dealt with next
if hyper.startswith("#http"):
        final_set.add(hyper[1:])

#This stage deals with the relative links that were left above by converting them to absolute urls
elif hyper.startswith("/"):
        final_set.add ("https://www.census.gov" + hyper)
```

Task D:

All duplicate links in the data end with either .gov or .gov/. This is relatively easy to deal with by simply converting all links that end in .gov to .gov/. Finally, this prevents duplicates as a set in python is not able to have duplicate entries. The program ensures that there are no duplicated links in the output file by using the following code in the screenshot:



```
#Here we add a '/' to all url's that end with .gov, combined with a set not being able to contain duplicates,
#all duplicates are now taken care of
elif hyper.endswith(".gov"):
        final_set.add (hyper + "/")
```

Task E:

Python code used to extract all the unique web links from the HTML code of the “Current Estimates” is in the file C996.py.

Task F:

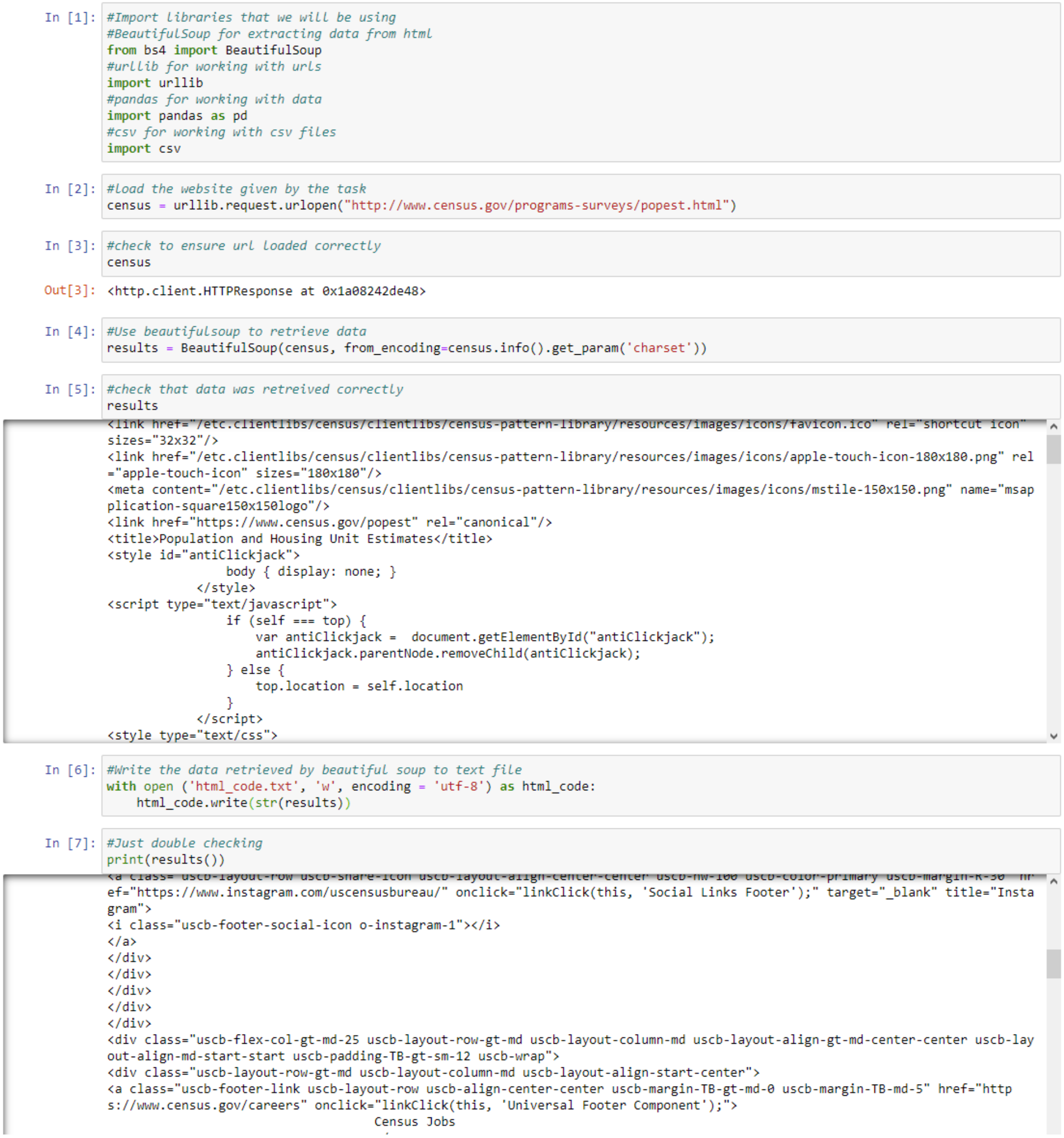
The HTML code of the “Current Estimates” scraped at the time when the scraper was run is html_code.txt.

Task G:

The CSV file created by my script is in the file unique_websites.csv.

Tash H:

Below are the screenshots I took of my code running successfully.

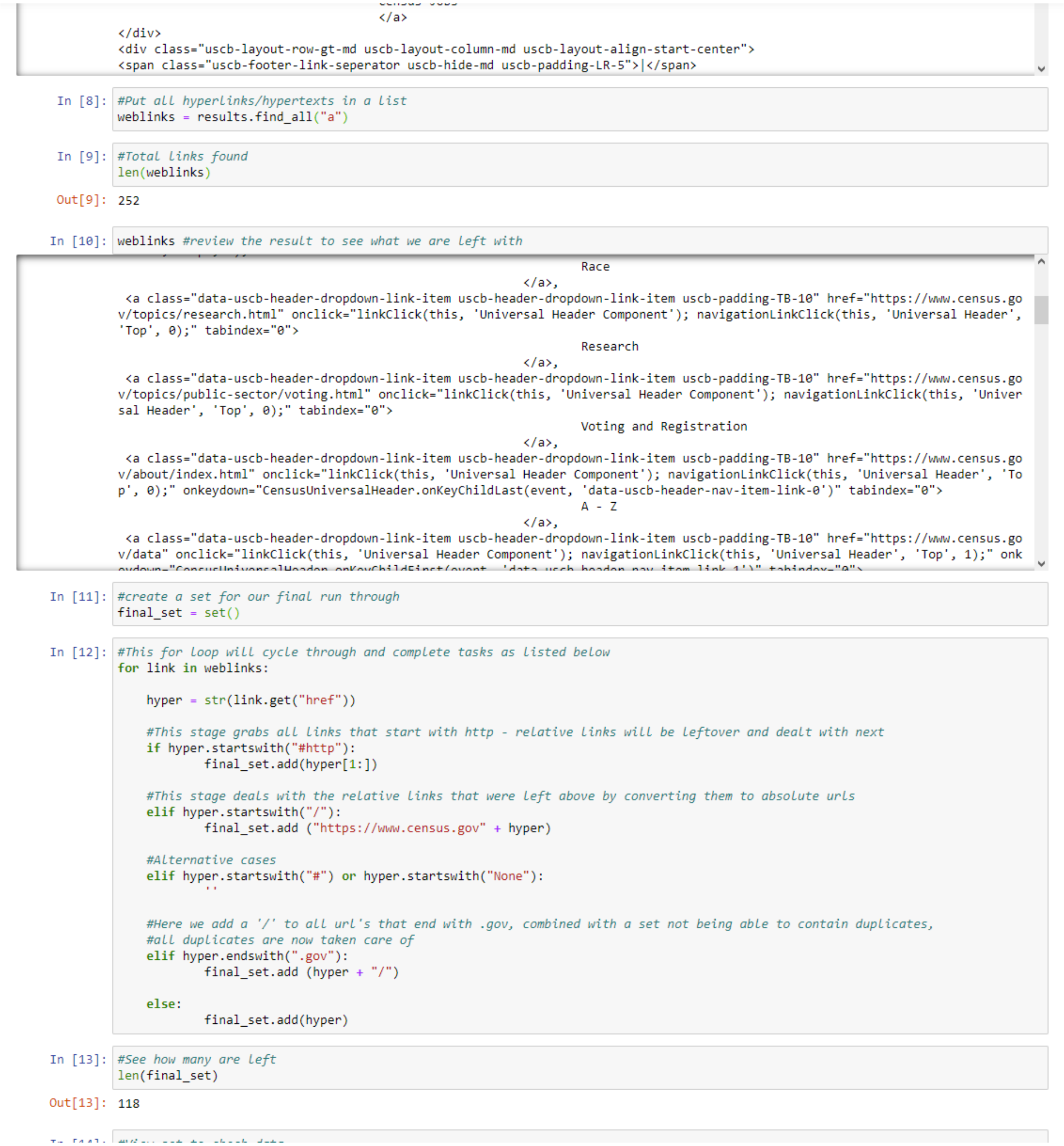


```
In [1]: #Import libraries that we will be using
        #BeautifulSoup for extracting data from html
        from bs4 import BeautifulSoup
        #urllib for working with urls
        import urllib
        #pandas for working with data
        import pandas as pd
        #csv for working with csv files
        import csv

In [2]: #Load the website given by the task
        census = urllib.request.urlopen("http://www.census.gov/programs-surveys/popest.html")

In [3]: #check to ensure url loaded correctly
        census

Out[3]: <http.client.HTTPResponse at 0x1a08242de48>

In [4]: #Use beautifulsoup to retrieve data
        results = BeautifulSoup(census, from_encoding=census.info().get_param('charset'))

In [5]: #check that data was retreived correctly
        results
```

```
<link href="/etc.clientlibs/census/clientlibs/census-pattern-library/resources/images/icons/favicon.ico" rel="shortcut icon"
sizes="32x32"/>
<link href="/etc.clientlibs/census/clientlibs/census-pattern-library/resources/images/icons/apple-touch-icon-180x180.png" rel
="apple-touch-icon" sizes="180x180"/>
<meta content="/etc.clientlibs/census/clientlibs/census-pattern-library/resources/images/icons/mstile-150x150.png" name="msap
plication-square150x150logo"/>
<link href="https://www.census.gov/popest" rel="canonical"/>
<title>Population and Housing Unit Estimates</title>
<style id="antiClickjack">
                body { display: none; }
            </style>
<script type="text/javascript">
                if (self === top) {
                    var antiClickjack =  document.getElementById("antiClickjack");
                    antiClickjack.parentNode.removeChild(antiClickjack);
                } else {
                    top.location = self.location
                }
            </script>
<style type="text/css">
```

```
In [6]: #Write the data retrieved by beautiful soup to text file
        with open ('html_code.txt', 'w', encoding = 'utf-8') as html_code:
            html_code.write(str(results))

In [7]: #Just double checking
        print(results())
```

```
<a class="uscb-layout-row uscb-share-icon uscb-layout-align-center-center uscb-hw-100 uscb-color-primary uscb-margin-R-30" hr
ef="https://www.instagram.com/uscensusbureau/" onclick="linkClick(this, 'Social Links Footer');" target="_blank" title="Insta
gram">
<i class="uscb-footer-social-icon o-instagram-1"></i>
</a>
</div>
</div>
</div>
</div>
</div>
<div class="uscb-flex-col-gt-md-25 uscb-layout-row-gt-md uscb-layout-column-md uscb-layout-align-gt-md-center-center uscb-lay
out-align-md-start-start uscb-padding-TB-gt-sm-12 uscb-wrap">
<div class="uscb-layout-row-gt-md uscb-layout-column-md uscb-layout-align-start-center">
<a class="uscb-footer-link uscb-layout-row uscb-align-center-center uscb-margin-TB-gt-md-0 uscb-margin-TB-md-5" href="http
s://www.census.gov/careers" onclick="linkClick(this, 'Universal Footer Component');">
                                    Census Jobs
```



```
                                         Census Jobs
                                    </a>
         </div>
         <div class="uscb-layout-row-gt-md uscb-layout-column-md uscb-layout-align-start-center">
         <span class="uscb-footer-link-seperator uscb-hide-md uscb-padding-LR-5">|</span>
```

```
In [8]: #Put all hyperlinks/hypertexts in a list
        weblinks = results.find_all("a")
```

```
In [9]: #Total links found
        len(weblinks)

Out[9]: 252
```

```
In [10]: weblinks #review the result to see what we are left with
```

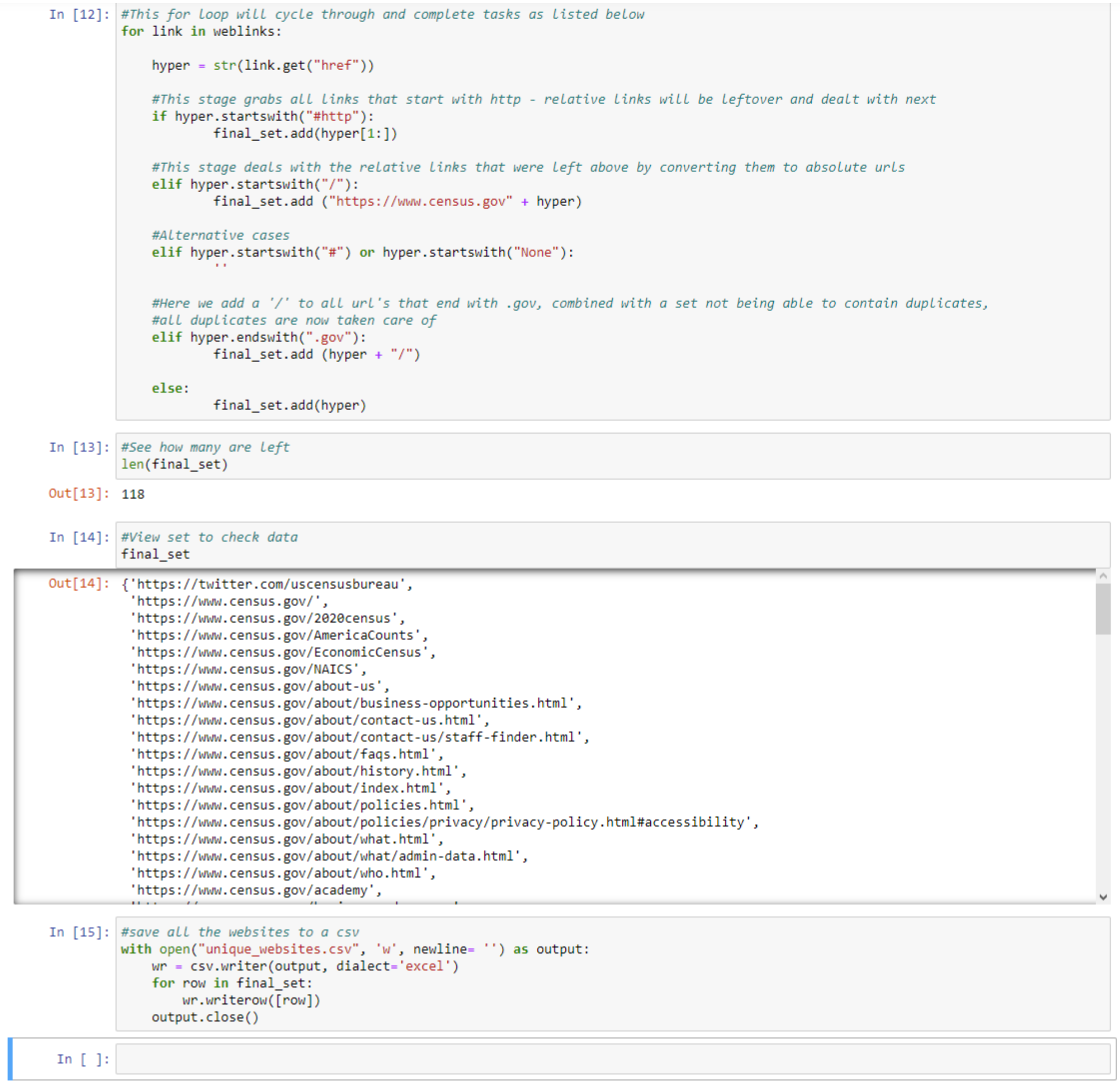
```
                                                        Race
                                               </a>,
 <a class="data-uscb-header-dropdown-link-item uscb-header-dropdown-link-item uscb-padding-TB-10" href="https://www.census.go
v/topics/research.html" onclick="linkClick(this, 'Universal Header Component'); navigationLinkClick(this, 'Universal Header',
'Top', 0);" tabindex="0">
                                                        Research
                                               </a>,
 <a class="data-uscb-header-dropdown-link-item uscb-header-dropdown-link-item uscb-padding-TB-10" href="https://www.census.go
v/topics/public-sector/voting.html" onclick="linkClick(this, 'Universal Header Component'); navigationLinkClick(this, 'Univer
sal Header', 'Top', 0);" tabindex="0">
                                                        Voting and Registration
                                               </a>,
 <a class="data-uscb-header-dropdown-link-item uscb-header-dropdown-link-item uscb-padding-TB-10" href="https://www.census.go
v/about/index.html" onclick="linkClick(this, 'Universal Header Component'); navigationLinkClick(this, 'Universal Header', 'To
p', 0);" onkeydown="CensusUniversalHeader.onKeyChildLast(event, 'data-uscb-header-nav-item-link-0')" tabindex="0">
                                                        A - Z
                                               </a>,
 <a class="data-uscb-header-dropdown-link-item uscb-header-dropdown-link-item uscb-padding-TB-10" href="https://www.census.go
v/data" onclick="linkClick(this, 'Universal Header Component'); navigationLinkClick(this, 'Universal Header', 'Top', 1);" onk
eydown="CensusUniversalHeader.onKeyChildFirst(event, 'data-uscb-header-nav-item-link-1')" tabindex="0">
```

```
In [11]: #create a set for our final run through
         final_set = set()
```

```
In [12]: #This for loop will cycle through and complete tasks as listed below
         for link in weblinks:

             hyper = str(link.get("href"))

             #This stage grabs all links that start with http - relative links will be leftover and dealt with next
             if hyper.startswith("#http"):
                     final_set.add(hyper[1:])

             #This stage deals with the relative links that were left above by converting them to absolute urls
             elif hyper.startswith("/"):
                     final_set.add ("https://www.census.gov" + hyper)

             #Alternative cases
             elif hyper.startswith("#") or hyper.startswith("None"):
                     ''


             #Here we add a '/' to all url's that end with .gov, combined with a set not being able to contain duplicates,
             #all duplicates are now taken care of
             elif hyper.endswith(".gov"):
                     final_set.add (hyper + "/")

             else:
                     final_set.add(hyper)
```

```
In [13]: #See how many are left
         len(final_set)

Out[13]: 118
```

```
In [14]: #View set to check data
```



```
In [12]: #This for loop will cycle through and complete tasks as listed below
         for link in weblinks:

             hyper = str(link.get("href"))

             #This stage grabs all links that start with http - relative links will be leftover and dealt with next
             if hyper.startswith("#http"):
                     final_set.add(hyper[1:])

             #This stage deals with the relative links that were left above by converting them to absolute urls
             elif hyper.startswith("/"):
                     final_set.add ("https://www.census.gov" + hyper)

             #Alternative cases
             elif hyper.startswith("#") or hyper.startswith("None"):
                     ''


             #Here we add a '/' to all url's that end with .gov, combined with a set not being able to contain duplicates,
             #all duplicates are now taken care of
             elif hyper.endswith(".gov"):
                     final_set.add (hyper + "/")

             else:
                     final_set.add(hyper)

In [13]: #See how many are left
         len(final_set)

Out[13]: 118

In [14]: #View set to check data
         final_set

Out[14]: {'https://twitter.com/uscensusbureau',
          'https://www.census.gov/',
          'https://www.census.gov/2020census',
          'https://www.census.gov/AmericaCounts',
          'https://www.census.gov/EconomicCensus',
          'https://www.census.gov/NAICS',
          'https://www.census.gov/about-us',
          'https://www.census.gov/about/business-opportunities.html',
          'https://www.census.gov/about/contact-us.html',
          'https://www.census.gov/about/contact-us/staff-finder.html',
          'https://www.census.gov/about/faqs.html',
          'https://www.census.gov/about/history.html',
          'https://www.census.gov/about/index.html',
          'https://www.census.gov/about/policies.html',
          'https://www.census.gov/about/policies/privacy/privacy-policy.html#accessibility',
          'https://www.census.gov/about/what.html',
          'https://www.census.gov/about/what/admin-data.html',
          'https://www.census.gov/about/who.html',
          'https://www.census.gov/academy',

In [15]: #save all the websites to a csv
         with open("unique_websites.csv", 'w', newline= '') as output:
             wr = csv.writer(output, dialect='excel')
             for row in final_set:
                 wr.writerow([row])
             output.close()

In [ ]:
```

Task I:

Sources:

Crummy. Retrieved from https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Coffee Cup. Retrieved from https://www.coffeecup.com/help/articles/absolute-vs-relative-pathslinks/