The purpose of this program was to explore an assortment of web scraping tools from the BeautifulSoup library to achieve a variety of different tasks involving manipulation of data supplied from a Wikipedia page containing recorded populations of Canadian provinces.

The requests library was used to retrieve the specified webpage to store in the variable “response”. Response.text was then used to initialize a BeautifulSoup object, and parsed using “html.parser”. This stored the webpage in a tree-like structure.

Important tables were then retrieved from this through calling the find_all() function to look for tables with the ‘wikitable’ class, AKA table elements on Wikipedia specifically used for displaying informational data. This list was then stored into the variable ‘tables’.

To contain all the tables data within a dictionary, first all time-period headers were retrieved (all headers from the tables were retrieved excluding ‘Name’). Then, each row was iterated through, and from each row the cells were extracted. The corresponding name of the province for a row of cells was retrieved and any citation note added to it was removed through regex. If the name was not already contained within the dictionary, it was added and given a list of blank values in correspondence to the list of time periods retrieved. For all other data, it was stored into its corresponding space by province/time period, replacing a blank space.

Duplicate time periods are removed while still maintaining order. This is then used to format the dictionary in a way that can be cleanly converted into a dataframe. The first column of the dataframe contains the names of all provinces and the rest of them contain corresponding values for the population of each province during a specific time period (if there is no specified data, there is only a blank space).

<h2> elements were then found by calling find_all(“h2”). The list obtained from the function call was then iterated through, and the text attribute of each item was printed (print(h2.text)).

To obtain all hyperlinks within the tables and insert them into a list, the tables variable was iterated through. Then, for each link found in tables.find_all(“a”), the list hyperlinks appended the value of ‘href’ using link.get(‘href’). The resulting list was then printed out.

The first step taken in traversing through the different links as contained in hyperlinks was first removing duplicates from said list so that the same webpage wasn’t downloaded twice. The list was then iterated through. For each link, it was first checked whether or not the link began with a # symbol, as this did not lead to different webpages but rather same page notes; if this was the case, the link was skipped over. If not, however, the link was appended to the website domain name (https://en.wikipedia.org) to create a full working url. Next, the webpage was retrieved using requests.get() and then parsed using BeautifulSoup, similar to the methodology used to retrieve the original webpage from the start of the program.

Each retrieved page was then binary written to an HTML file using BeautifulSoup’s UTF-8 encoder. This allowed the HTML code of each webpage to remain preserved in a way that still functioned. Each file was saved to the ‘webpages’ directory and named after the title of the respective webpage they were downloaded from.