**Python functions results 50 most common words on any website (main website along with connected ones/links/documents), and corresponding frequency.**

Write the result to an output file.

1. Start crawling from any website.

2. Never visit the same page more than once.

3. Visit pages that are only WITHIN the main page domain – the url starts with the beginning of the absolute URL. Do not visit external sites.

4. When you process the ‘data’ (processed by the ‘handle\_data(data)’ function defined in the Python HTMLParser class, which is inherited in your ‘Collector’ class; assuming you used the code shown in the lecture PPT), convert all data to lower case.

5. In the 50 most common words, DO NOT include stopwords (e.g. ‘the’, ‘a’). Stopwords text file provided.

**Hints:**

* After creating an absolute url (in Collector), if the final url contains either ‘mailto’ or ‘img’ or ‘'course-evaluations', do NOT traverse the link, if do your code will error.
* In Python HTMLParser, when feed() is called, the order of the tag/data detection sequence is: handle\_starttag() | handle\_data() | handle\_endtag()

When access the data returned from handle\_data(), which you will have to override in your Collector class, the data was from irrelevant/unwanted sections, such as a section started by the tag <script>, <meta>, <script>, <code>. Do NOT process data from those sections.

Store the tag that was detected in the handle\_starttag(). Then when handle\_data() is invoked next (automatically), check the tag you stored for the data section, and if the tag was one of the unwanted tags, you ignore the data extracted from the (tagged) section. List of unwanted tags: ['script', 'noscript', 'input', 'meta', 'title', 'style', 'form'].

Be sure to remove punctuations, such as ‘,’, ‘.’, ‘:’, ‘?’. ‘!’, from the tokens in data. Note that there could be any number of punctuations (not just one) given to a word, such as “okay?!!” and “<-good”.