Web Crawler

In this exercise you will implement a simple web crawler. The exercise has to be implemented in Python or Java. You may use open-source libraries, provided that they do not implement web crawling.

Running the crawler

The web crawler should be executed with two arguments - the URL of the root page (URL as a string i.e. http://www.wikipedia.org) and the recursion depth limit (a positive integer).

Fetching a URL

Each URL given to the fetcher should be examined for its mime type and processed only if its mime type is `text/html`.

The processed URLs should be downloaded to disk (to a directory structure of your choosing) and named as the URL it originated from, up to characters that cannot be included in file names.

All the static links that are contained in the page (i.e. that are defined in `<a>` tags of the original source, not including links that may be dynamically added by scripts) should be extracted from the page and inserted to the crawler for fetching, if the depth limit is not exceeded. A depth limit of 1 means that only the URL given to the crawler upon execution will be processed.

Processing the page

For each page, calculate the ratio of same-domain links that it contains, as a number in [0, 1]. For example, say the page at http://www.foo.com/bar.html contains the following links:

- *http://www.foo.com/a.html*
- *http://www.foo.com/b/c.html*
- http://baz.foo.com/
- http://www.google.com/baz.html

Then the ratio will be 0.5, since the two italic links are same-domain links and the rest are not. Note that same-domain links must have a common prefix, including subdomain.

Persistency and caching

The crawler should be able to continue crawling without restart even if it had been stopped (by, for example, terminating the process). It is allowed to have a small amount of work re-done by the crawler, but the vast amount of work should not be repeated.

Additionally, after the crawl is completed, the crawler should support running at a later time, and it should avoid, as reasonably possible, re-downloading pages that were already downloaded and were not updated since the last crawl.

Output

The crawler should output a TSV (tab separated values) file containing at least three columns - the URL of the page downloaded, its depth in the links tree and its same-domain links ratio, for example:

```
url   depth ratio
https://news.ycombinator.com    1     0.1
https://nodejs.org/en/blog/release/v4.0.0/ 2     0.65
https://news.ycombinator.com/item?id=10187596   2     0.8
https://nodejs.org/en/     3     0.38
```