Steven Krenn, Wendy Wei

**Instruction**

To run the program you will need the dependencies, re, CSV, requests, and BeautifulSoup4. Whenever you run the program it will develop a text file called “output.txt” This file is used in the html file construction. It then constructs the html files. Then it constructs the inventor html files. To access the home page, go to index.html. To access an inventor’s page either use the hyperlinks in the pages, or use the format: “lastname-firstname.html”. For example you could go to “feinberg-adam.html”

**Objective**

This program processes [U.S. patent application](http://appft.uspto.gov/netacgi/nph-Parser?Sect1=PTO2&Sect2=HITOFF&u=%252Fnetahtml%252FPTO%252Fsearch-adv.html&r=0&p=1&f=S&l=50&Query=aanm%252F%2522carnegie+mellon%2522+AND+PD%252F4%252F1%252F2016-%253E6%252F30%252F2016&d=PG01) data to produce a web-based “map” that can be used to understand the topics and subtopics of research, along with the researchers/authors, who are engaged in those topic areas.  The program would process, as web-scraped input, the inventory data (which includes patent information and creators) that are seeking a U.S. patent for a discovery.  Your program would select topic and subtopics from the input, link those keywords to the creator(s) and patent title(s), and then produce a useful web interactive means of interrogation for querying.

**Methodology**

We first used the requests and BeautifulSoup libraries on this project. The requests library requests a website and calls down all its source html code. BeautifulSoup parses the html file in to a more readable format. We used BeautifulSoup to output all the hyperlinks of the file. We then processed the list in to a workable format for iteration. We then iterated over the list and wrote the contents of each patent website’s text to a master file. After we have the master output.txt file, we parsed the text file and create a dictionary of keywords and how many times they show up in the output.txt file. We then sort dictionary from highest frequency of occurrence to lowest. We then output that sorted dictionary to the index.html file for output.

To link each keyword to patent(s), we first split the number lines into 15 intervals. Then we use line indexes of each keyword to locate where this keyword belongs to. To provide basic information for each patent, we used a dictionary which has line intervals as keys and patent application name(s), application number(s), and inventor(s) as values.

To link each inventor to patent(s), we first stored a list of inventors for each patent. Next, we looked for every inventor’s associated patent(s) and return basic information for that patent(s).

This tool also allows user to input a ‘txt’ file and extract keywords out of it and then output it in an html file. Before extracting the keywords, this tool first performs text analysis using the delete junk words and substituting miss-spelling words. Then it pulls out a list of keywords and order them so the most frequent word appears first. Finally, this tool output all the keywords into an html file.

**Scope**

This tool can be applied to any other patent files which has a similar format of current input file. It can be applied to a different search of 15 links from the input website.