Stephen Knapper
Stephen.a.knapper@wmich.edu

A. Description

The goal of this project was to create a "search engine."
To do this, it was my understanding that I was to take a file of URLs and keywords, read it and put those keywords into a hashmap with the related URLs.

B. Algorithms

In my first hashmap I used the 'buckets' hashmap. I took the length of the keyword, multiplied it by 6 and modded the size of the hashmap.
In my second and best hashmap, I used linear probing. If the index was not empty, I probed forward by adding 7.
To hash the value in my second map, if the value was a string, I looped through each character in the keyword, took its ASCII value and squared it and added it to a total. Then I multiplied that value by a large prime number and modded it to fit in my hashmap.

C. Theoretically Derived Complexities

The theoretical complexity of a hashmap is O(1).
Increasing the size of the hashmap, should it become too full, will be O(n)
Searching the hashmap for a value should be O(1)

D. Observed Time Complexities

Adding a value to the hashmap takes O(1), unless the map is too large, then resizing takes O(n).
Searching the hashmap to find a value is O(1). Adding those URLs to the stack/list takes n number of URLs related to the keyword.

E. Theoretical Vs. Empirical

The only part of the code that doesn't match up with the theoretical big-oh, is searching. This is due to the nature of the assignment. When searching for the AND operator, I have to search through a list to ensure that the URL is related to both keywords.