Text Retrieval & Search Engine (CP423)

Assignment 1

## Authors

Brandon Parker – 191593730

Samson Goodenough - 190723380

## Text Preprocessing

Text preprocessing involved several steps to prepare the corpus for analysis. Firstly, we traversed through each file and performed normalization and tokenization. This entailed retaining alphanumeric characters and spaces while substituting newline characters with spaces. Subsequently, we utilized the NLTK library to tokenize the characters into an array. Finally, we removed stop words using the stop words list provided by NLTK.

## Inverted Index

We opted to employ a Python dictionary to construct the inverted index due to its inherent hashing implementation, which minimizes lookup times. Utilizing a dictionary also enhances code readability, increasing comprehension for anyone examining the process. In the inverted index, words served as the lookup keys, while associated with them were lists of document IDs where each word occurred.

## Query Support

Queries are submitted by the user after loading the inverted index. First, the user inputs the number of queries; then, their chosen keywords; and finally, their operators to be inserted into the keywords. For example, “cat, dog, bird” and “AND OR NOT” would become “cat AND dog OR NOT bird”. After inputting the query, the program first strips the input of any non-alpha characters, casts them to lowercase, and filters out stop words. Then the operators are stripped, split into an array and checked for any invalid operators which will be removed. Finally, the operators are grouped for the “OR NOT” and “AND NOT” cases, and alternately sliced into the array of keywords reading the inputs from left to right. This means any invalid inputs or extra inputs will be truncated at the end of the query.

## System Evaluation

During the evaluation phase, we generated comprehensive results encompassing the count of matched documents, required comparisons, and a catalog of retrieved documents. We tracked the number of comparisons necessary for each operation as requested by the user, offering a benchmark for assessing our algorithm's efficiency. Additionally, we presented the user with the document ID and corresponding file names of the retrieved documents, facilitating ease of review.

## Conclusions

This assignment offered a comprehensive exploration into various aspects, including file I/O, normalization, and list comprehension, among others. Exploring the NLTK library was interesting as it introduced us to a new library. As always, handling user inputs posed challenges, requiring anticipation of unexpected scenarios that could arise. The most intriguing aspect centered around the depth of normalization possibilities. While removing stop words is a common step, the process opens doors to considerations such as character retention, prefix and suffix adjustments, and other avenues for enhancing the search algorithm. It's a realm filled with diverse paths for optimization and improvement.