Let's Search
Design Document
Sam Hunter, Morgan Monzingo
11.17.14

Our team consists of Samuel Hunter and Morgan Monzingo.

UserInterface (Morgan) : This class handles the interaction between the user and the search engine. This allows users to choose between the thee main search engine modes, maintenance, interactive and stress test.

DocumentParser (Morgan and Sam) : This class uploads the xml file, breaks that file into specific pages. Then it is able to read in the specific content, remove stop words and stem each word. This class can also send the pages to the IndexHandler and save the pages.

Page (Morgan and Sam): The page class takes all of the information from each specific page and stores it as the appropriate variables.

QueryProcessor (Morgan): This class takes the query entered by the user and interprets what page objects the user is searching for. This class interacts mainly with the Query and IndexHandler class.

Query (Morgan): This class stores all of the data associated with the user's query.

IndexHandler (Sam): IndexHandler can add page object to the index, search the index for specific queries and delete the index.

AVLTree (Sam): AVLTree is a subclass of indexHandler. This class is responsible for creating, manipulating and balancing our AVL tree index structure.

HashTable (Sam): HashTable is a subclass of indexHandler. This class is responsible creating and managing the hash table index structure.

Node (Sam): Node is a class that makes up AVL trees and hash tables, it's responsible for holding a small portion of the larger data set.

HashNode (Sam): HashNode is a subclass of the Node class. It is responsible for holding a portion of the larger data set specific to the rules of the hash table.

AVLNode (Sam): AVLNode is a subclass of the Node class. It is responsible for holding a portion of the larger data set specific to the rules of the hash table.

Stress Test Commands:

Parse x documents : Parse however many documents given, then throw away data.
Search "string" x times: Search index for string given number of times.
Parse and index (AVL Tree): Parse however many documents given and save them to the AVL tree index.

Parse and index (Hash Table): Parse however many documents given and save them to the hash table index.
Save index to file (AVL Tree): Saves AVL tree index to file.
Save index to file (Hash Table): Saves hash table index to file.
Parse index and save (AVL Tree): Parse however many documents given and save them to the AVL tree index, then save to the disk.
Parse index and save (Hash Table): Parse however many documents given and save them to the hash table index, then save to the disk.