(1) *Input*
Command Input

./indexer [folder] [file to create]

./indexer [folder] [file to create] [file to test] [new output test file]

example command input
./indexer data/ index.dat
./indexer data/ index.dat index.dat new_index.dat

[folder]
Requirement: The folder must exist, and must contain HTML files numbered from 1 to n, where n is the number of files in the folder.
Usage: The indexer will index these files.

[file to create]
Requirement: None
Usage: the indexer will write over this file with the frequency, and location of all of the words found in the files provided by the indexer.

[file to test]
Requirement: Must exist, and must be of the format 'word 1 900 12'
Usage: is read by indexer, and recreated in a new file

[new output test file]
Requirement: None
Usage: same as [file to create]

(2) *Output*
The indexer will make a file containing all of the words found in each of the html files in the file provided. The output will be of the format:

[Word] [number of documents] [document number] [frequency of word]

The file will be in Alphabetical order. There is no order on the documents.

(3) *Data Flow*
Each word is stored in 2 places, an ordered linked list which contains all of the words, and a hash table of linked lists, which also contains a linked list of all of the documents and the frequency of the words on those documents.

(4) *Data Structures*
As mentioned above, we will need a singly linked-list, and a HashTable.

(5) *Indexer Pseudocode*

```
// Check first 3 command line arguments

// initialize hashtable, and linked-list

// Open folder

// While there are documents
        // get document

        // While there are words
                // Get Word

                // Add Word to structures

// print out data structures

// Check next 2 command line arguments

        // open second file

        // while file has lines
                //add line to data structures

                //print out data structures
```

Testing:
I tested bad inputs, and each of the three levels of input that we got from the crawler

bad inputs:

```
./indexer dat/ index.dat            // dat/ doesn't exist
./indexer data                      // not enough arguments
./indexer data index.dat asdf       // should run fine, until end when it complains
                                    //about an
                                                    // incorrect number of arguments
./indexer data index.data README new_index.dat// incorrect file format for input
```