Plagarism Checker

Usage

```
make
./bin/plagChecker <test_file> <corpus_folder>
```

If documents have a single theme (e.g. essays on the same topic), use the `same-theme` option:

```
./bin/plagChecker <test_file> <corpus_folder> same-theme
```

Working

The basic measure of similarity used to detect plagarism is the **cosine similarity**. This is because the goal was to check for text similarity and not the program control flow or order. Also, cosine similarity works well for documents or uneven or large sizes. Cosine similarity works by creating two *word vectors* for both the documents to be compared and then taking the cosine of the tow vectors. Each unique word in a document accounts for a unique dimension in the vector with value equal to the number of occurences or the *frequency* of that word.
Cosine similarity formula:

```
cosine similarity = (x . y) / mod(x) * mod(y)
(where x and y are word vectors)
```

For documents having a single theme or topic, cosine similarity with **term frequency and inverse document frequency** or **TF-IDF** can be used via the `same-theme` option. This can be benefitial in tasks like detecting plagarism in essays written on the same topic. For example, if there are 20 essays on the topic "cricket", then even if none of the essays are copied, the words like "cricket", "bat", "stadium" etc. will be present with a high frequency in each document and simple cosine similarity can give vague results. So, the *weight* of such words is reduced relative to other unique words in the calculation of similarity in the TF-IDF method.

Time and space complexity

If the average number of words in the test file and corpus files is `n` and the number of corpus files is `m`,
Time complexity for iterating over words and checking for uniqueness = `O(n^2)`
Time complexity for calculating cosine similarity = `O(n)`
Space complexity for storing words of an individual file = `O(n)`
Space complexity for storing all words = `O(m * n)`

So, time and space complexity of the program:
Time complexity for cosine similarity = `O(m * n^2)`
Space complexity for cosine similarity = `O(n)`
When run with `same-theme` option:
Time complexity for cosine similarity with TF-IDF = `O(m * n^2)`
Space complexity for cosine similarity with TF-IDF = `O(m * n)`

Implementation

Implemented in C.
Used a generic linked list for storing the list of words. (See `include/LinkedList.h` or `src/LinkedList.c`)
Used a `Word` data-type to represent unique words. (See `include/Word.h` or `src/Word.c`)

```
typedef struct Word
{
    char word_text[MAX_WORD_SIZE];
    int frequency;  // term frequency
    double idf;     // inverse document frequency
} Word;
```

Set the `DEBUG` macro in `src/plagChecker.c` to 1 for easy debugging.

Author

Soham Gaikwad, 2018CS10394