Programming Assignment #5

Copycat Catcher

CS 3358.253, Spring 2015
Instructor: Jill Seaman

Due: Wednesday, 4/15/2015 (upload electronic copy by 1:30pm)

Problem:

You will be writing **part** of a tool that could be used to catch plagiarists. The goal of the tool is to determine similarities between documents from a large set to find out if plagiarism is going on within the group.

For this assignment you will write a program that will be able to compare two documents and indicate how many n-word sequences they have in common.

Your program should take as input the name of two files in the working directory, and the value for n (the number of words in a sequence). Your program can prompt the user to enter this information. Then your program should output the number of n-word sequences that the files have in common (ignoring duplicates in either file).

Here is the output from my program. All output is optional except the number of sequences in common.

```
Please enter name of the first file to analyze:
sm_doc_set/catchmeifyoucan.txt
Please enter name of the second file to analyze:
sm_doc_set/ecu201.txt
Please enter the number of words per sequence:
5
***sm_doc_set/catchmeifyoucan.txt
***sm_doc_set/ecu201.txt
word count file1 = 1010
word count file2 = 1291
sequences in common: 289
```

Part I:

The first step is to produce a list of **all the n-word sequences** that occur in a text document. A **word** is a sequence of characters containing no whitespace (**whitespace** includes spaces, tabs, and newlines). An **n-word sequence** is just a list of n words that occur in that order in the document (each separated by a single space).

For example, if the document starts this way:

```
Death of a Salesman:
In the play, Arthur Miller’s Death of a Salesman: Willy Loman, a sympathetic
salesman and despicable father who’s “life is a casting off” has some traits that
match Aristotle’s views of a tragic hero. Willy’s series of “ups and downs” is identical
to Aristotle’s views of proper tragic figure; a king with flaws. His faulty personality,
the financial struggles, and his inability are three substantial flaws that contribute to
his failure and tragic end.
```

The list of 6-word sequences would start out this way:

```
Death of a Salesman: In the
of a Salesman: In the play,
a Salesman: In the play, Arthur
Salesman: In the play, Arthur Miller’s
In the play, Arthur Miller’s Death
the play, Arthur Miller’s Death of
play, Arthur Miller’s Death of a
Arthur Miller’s Death of a Salesman:
Miller’s Death of a Salesman: Willy
Death of a Salesman: Willy Loman
...
```

Do not simply break the file up into n-word segments that, when appended will re-create the file. In other words, this list is incomplete:

```
Death of a Salesman: In the
play, Arthur Miller’s Death of a
Salesman: Willy Loman, a sympathetic salesman
. . .
```

Also note that newlines (and tabs) from the input file should not be included in the n-word sequence. For Part II, each n-word sequence should be stored in a single string.

Part II:

Use a hash table to compute the results for your program. Implement a **hash table** for containing strings using the class declaration given below. Implementing the hash table is a requirement for this programming assignment!! It is not a template, so you should put your function definitions in hashtable_3358.cpp. You may use member variables of any data type for the hash table (array or vector, for example).

```
class HashTable_3358 {
public:
    /****************************
     constructor: Initializes hash table to a predetermined size.
     ****************************/
    HashTable_3358 ( );

    /****************************
     makeEmpty: Removes all the items from the hash table.
     ****************************/
    void makeEmpty();

    /****************************
     find: Returns true if s is in the hash table, else false.
     ****************************/
    bool find(string s) const;

    /****************************
     insert: Adds the string to the hash table, if it is not
     already there. (If the hash table becomes 50% full, it rehashes.)
     ****************************/
    void insert (string s);
}
```

Your hash table should also include private functions to handle the following tasks:

- hash(string) : calculates the position of the string in the table
- probing: uses **quadratic** probing to resolve collisions
- rehashing: doubles the size of the array (at least) when it becomes 50% full.

NOTES:

- I will be putting a zip file of text files on the class website soon. You can use these files to test your tool (see the file called "catchmeifyoucan.txt").
- Write your own hashtable_test.cpp file to test your implementation before you start writing the main program. This will save you from headaches later.
- You may use the code at the end of the document to compute prime numbers:

Style: See the Style Guidelines document on the course website.

Logistics:

Please submit the following files in a single zip file. You can call it assign5_xxxxxx.zip, where xxxxx is your TX State NetID (your txstate.edu email id).

`hashtable_3358.h hashtable_3358.cpp driver.cpp`

Submit: an electronic copy only, using the Assignments tool on the TRACS website for this class.

```
bool HashTable_3358::isPrime(int number)
{     // Given:   num an integer > 1
      // Returns: true if num is prime, false otherwise.
      int i;
      for (i=2; i<number/2; i++)
      {
        if (number % i == 0)
        {
                return false;
        }
      }
      return true;
}

int HashTable_3358::nextPrime (int number) {
   int x = number+1;
   while (!isPrime(x))
       x++;
   return x;
}
```