HACETTEPE UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

BBM-203 PROGRAMMING LAB.

ASSIGNMENT 3

Subject: Linked Lists

Submission Date: 13.11.2017

Deadline: 27.11.2017

Programming Language: ANSI C

Operation System: Linux

Advisors: Asst. Prof. Dr. Burcu CAN, Asst. Prof. Dr. Adnan ÖZSOY, Assoc. Prof. Dr. Sevil ŞEN**,**

R. A. Pelin Canbay

1. Introduction / Aim

In this assignment, you are expected to design a document analysis program. The primary goal of this experiment is to get you practice on linked lists and matrices.

2. Background Information

Document analytics is the process of deriving information from text. The documents are examined to obtain information about authors by analyzing the texts. It is expected that an author's documents will be similar to each other. Therefore, author of a document can be identified using similarity between other documents.

The similarity is a measure of the resemblance between datasets; i.e., how similar or alike the sets are. The most commonly used datasets are terms when the similarity of documents is considered. Documents similarities can be found using similarity metrics based on the frequency of terms in documents.

In Information Retrieval approaches, the commonly used similarity measure is the Cosine Similarity. Cosine similarity is a measure of similarity between non-zero vectors of an inner product space that measures the cosine of the angle between them. When the documents are the subject, this measure is computed by representing datasets as vectors in term space and then comparing the angles formed by the vectors.

Given two vectors of terms, A and B, the cosine similarity is represented using a dot product and magnitude (Magnitude is defined as the length of a vector. The notation for absolute value [|| ||]is also used for the magnitude of a vector.) as follows:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

The cosine of 0^o means that cos(0) = 1 and that the vectors have identical directions, in other words, the corresponding datasets are completely similar to one another. The cosine of 90^o means that cos(90) = 0 and that the vectors are perpendicular, in other words, corresponding variables are uncorrelated. Consequently, if the cosine similarity is closer to 1, the similarity of vectors increases. In the case of information retrieval, the cosine similarity of two documents will range from 0 to 1, since the term frequencies cannot be negative. The angle between two vectors of term frequency cannot be greater than 90^o.

3. Experiment

The primary goal of this experiment is to make you acquainted with operations in linked lists like adding, deleting, sorting etc.

In this experiment, you are expected to write an application that calculates the document similarity and holds the pairs of words in a document. And you are also expected to use linked lists and matrices when performing this assignment.

You will design and implement an application program named ***Simon***, in ANSI C and compile it with gcc. You will also use the *make* utility and write a *makefile* [7] for your project.

The application is a document analysis program. Simon will get some commands and parameters as arguments from the console until the user sends –q parameter.

Example

–r bbm203/p3/d1.txt # read d1.txt form the given path

–a w2 3 d1.txt #add w2 with count 3 in d1.txt

–n2 d1.txt #get first 3 pairs of words that pass the most on the document

–d w1 d1.txt #delete w1 from d1.txt

–r bbm203/p3/d2.txt # read d2.txt form the given path

–s d1 d2.txt #return similarity of d1.txt and d2.txt

-q #quit the application

Details of example

With **–r** command, the application will read the given document word by word. In order to obtain words, you should stop reading on all non-alphanumeric characters. Also, you should hold all words and counts in a linked list for each read document.

With **–a** w2 3 d1.txt command and parameters, the application will add the word w2 and its count (3) to the list of d1 document.

With **–n2** d1.txt command and parameter, the application will return the first 3 pairs of words that pass the most on the document d1.txt. (Pairs of words means that two consecutive words. For example, the sentence “today the weather is so cold” has 5 pairs of words; today-the, the-weather, weather-is, is-so, so-cold) To get the pairs of words, you have to design **nested linked list**.

With **–d** w1 d1.txt command and parameters, the application will delete the node which holds the information of w1, from the linked list

With **–s** d1.txt d2.txt command and parameters, the application will return the cosine similarity of d1.txt and d2.txt. To simplify operations, the application will calculate just fist 10 words of the documents’ sorted list. It means that **there will be the maximum 20x2 (word x document) matrix for calculation of similarity**.

With **–q** command, the application will free to memory and close.

A sample application

A sample application is detailed below, so you can understand the solution of the problem better.

The contents of d1.txt and d2.txt are as follows.

d1.txt	The funct printf( ) and scanf( ) types, plus null-term and char strings. Char strings char and. Blah blah blah blah
d2.txt	This is testing for printf... This is testing for scanf... char strings and char. fgetc(file) to5 blah blah blah blah blah the and

-r d1.txt: With this command d1.txt document is read by the application. While reading, the application creates a linked list for the document and adds words to list with their pair. The final representation of this process is shown in Figure 1.

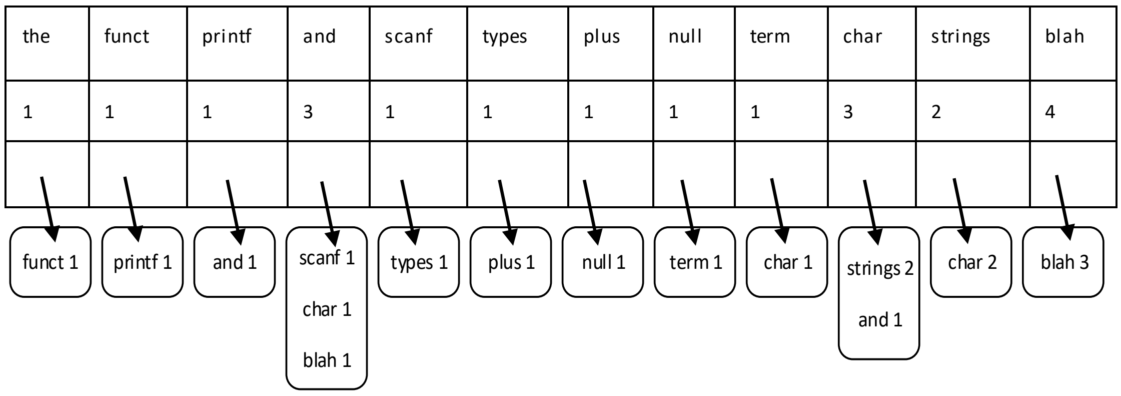


Figure 1. Linked list of words of d1.txt before sorting

You have to sort the linked list. You can sort the list while adding the words or after reading all text. The final representation of sorting is shown in Figure 2. (P.s. The bubble sort algorithm is used on this sample. You can use any sorting algorithm in your application.) There won't be any output.

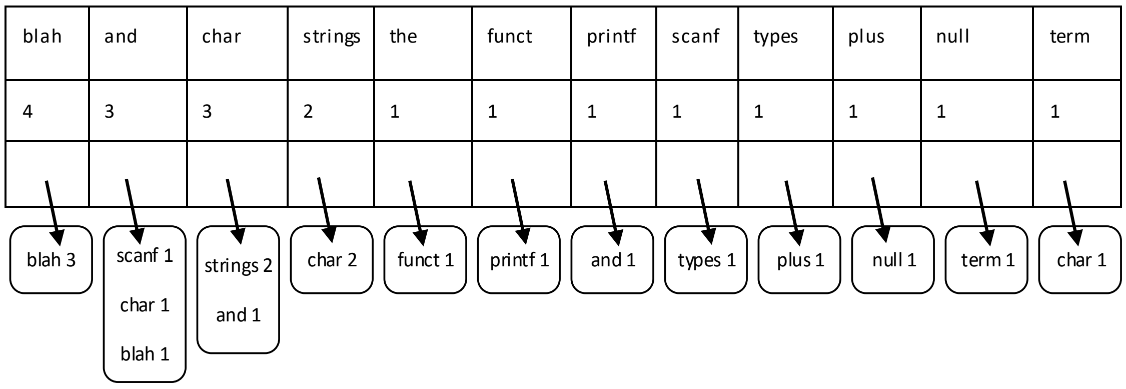
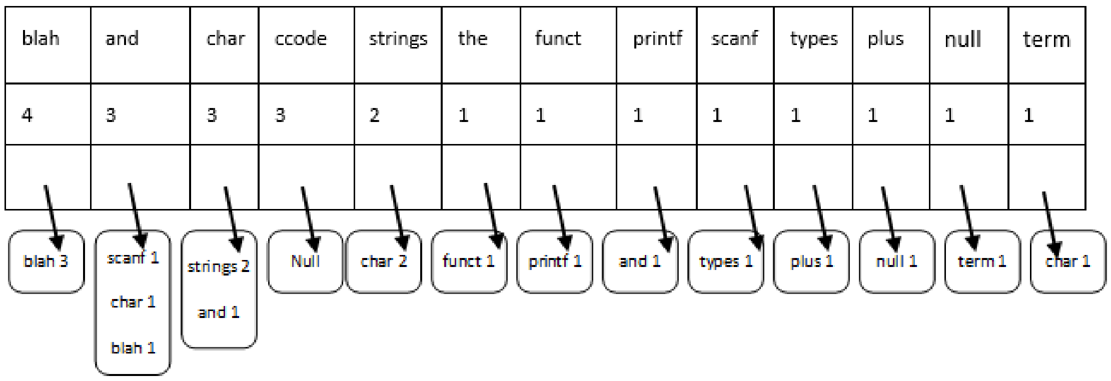


Figure 2. The linked list of words of d1.txt after applying bubble sort algorithm.

–a Ccode 3 d1.txt: the application will add the word Ccode and its count (3) to the sorted linked list of d1 document. The final representation of the linked list of d1.txt after adding the given word and its count is shown in Figure 3.



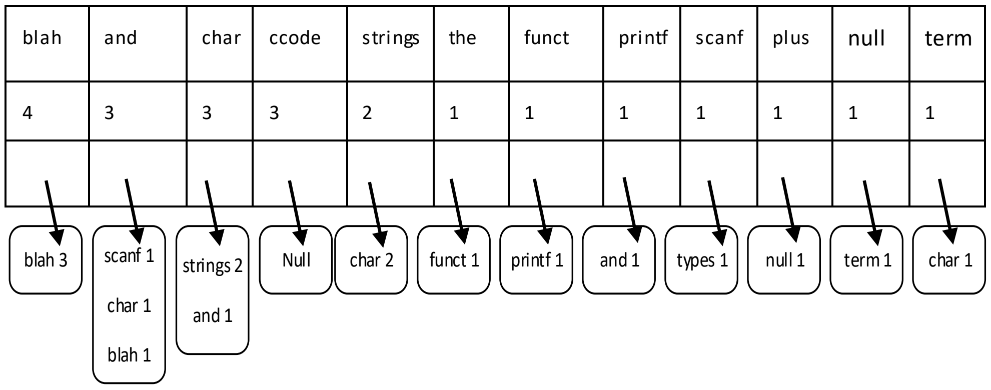
blah	and	char	ccode	strings	the	funct	printf	scanf	types	plus	null	term
4	3	3	3	2	1	1	1	1	1	1	1	1

Figure 3. The representation of the linked list after adding given word and its count

–n2 d1.txt: For this command, the application will print first 3 pairs of words that pass the most on the document d1.txt on the screen of the console.

Blah-blah, char-strings, strings-char

–d types d1.txt: For this command, the application will delete the node which holds the information of the given word ("types"). There won't be any output but you have to update the linked list of given document ("d1"). The representation of the linked list of d1.txt after delete operation is shown in Figure 4.



blah	and	char	ccode	strings	the	funct	printf	scanf	plus	null	term
4	3	3	3	2	1	1	1	1	1	1	1

Figure 4. The representation of the linked list after deleting the given word

-r d2.txt : d2.txt document is read by the application. The final representation of this process is shown in Figure 5.

blah	this	is	testing	for	char	printf	scanf	strings	fgetc	file	to5	the	and
5	2	2	2	2	2	1	1	1	1	1	1	1	1

Figure 5. The linked list of the document d2.txt after reading and sorting

–s d1.txt d2.txt: After getting this command the application will return the cosine similarity of d1.txt and d2.txt. First, you will get the first 10 words from the linked list of d1.txt and d2.txt. Then you will create a document-word matrix according to the number of different words you have. And then, you will put the words counts into the document-word matrix. The document-word matrix of this sample application is shown in Figure 6. Finally, you will apply cosine similarity on the matrix and will print the result of similarity on the screen of the console like below.

	blah	and	char	ccode	strings	the	funct	printf	scanf	plus	this	is	testing	for	fgetc
d1	4	3	3	3	2	1	1	1	1	1	0	0	0	0	0
d2	5	1	2	0	1	1	0	1	1	0	2	2	2	2	1

Figure 6. The document-word matrix of this sample application

Cosine Similarity of d1.txt and d2.txt is 0.660

Important Issues

- In order to obtain terms, you should stop reading on all non-alphanumeric characters.
- All terms in your list must be in lower case
- Be careful! If the magnitude of two documents is zero, there will be division by zero error when calculating cosine similarity. In this case, the result will be zero.
- **Test your program on "dev.cs.hacettepe.edu.tr" before submission**. Your submission will be compiled and executed on this machine and this machine only.
- **Projects without a proper Makefile won't be graded.**
- Your application will be controlled for possible memory leaks and you will get bonus points.

Assignment Report

- Report will be 30 points
- Please explain your design, data structures, and algorithms.
- Give necessary details without unnecessary clutter.
- Do a Big-O notation algorithm analysis of your functions
- Guide: ftp://ftp.cs.hacettepe.edu.tr/pub/dersler/genel/FormatForLabReports.doc

Last Remarks

- Regardless of the length, use **UNDERSTANDABLE** names to your variables, classes, and Functions
- Write **READABLE SOURCE CODE** block
  - These will cost you points.
- Deadline is 23:59 pm. No other submission method will be accepted.
- **Save** all your work until the assignment is graded.
- The assignment must be original, **INDIVIDUAL** work. Duplicate or very similar assignments are both going to be punished. General discussion of the problem is allowed, but **DO NOT SHARE** answers, algorithms or source codes.
- Copying without citing will be considered as cheating. Citing does not make it yours; cited work will get 0 from the respective section.

- You will submit your work from https://submit.cs.hacettepe.edu.tr/index.php with the file hierarchy as below:

This file hierarchy must be zipped before submission (Not .rar, only .zip files are supported by the system)

➔ <student id>.zip
    - Report
        - report.pdf
    - Source
        - *.c
        - *.h
        - Makefile

References

1- C The Complete Reference 4th Ed Herbert Schildt 2000
2- Garcia, E. (2015b). A Tutorial on Distance and Similarity. Retrieved from

   http://www.minerazzi.com/tutorials/distance-similarity-tutorial.pdf

3- C The Complete Reference 4th Ed Herbert Schildt 2000
4- Calculus II For Dummies, 2nd Edition, Mark Zegarelli
5- Fundamentals of Data Structures in C. Ellis Horowitz, Sartaj Sahni, and Susan Anderson-Freed, 1993
6- BBM 201 - Data Structures (Fall 2016) http://web.cs.hacettepe.edu.tr/~bbm201/
7- http://web.mit.edu/gnu/doc/html/make_3.html