Scores:			
	1)	5)	9)
	2)	6)	10)
	3)	7)	
	4)	8)	

COMP 4321 Search Engine for Web and Enterprise Data
Final Examination, Fall 2012
December 12, 2012
Time Allowed: 2 hours 15 mins

Name: _____ **Student ID:** _____

Note: Answer all questions in the space provided. Answers must be precise and to the point.

1. **[10]** Circle True or False in the following questions:

   T F False drops in a signature file are mainly due to information loss resulted from superimposition.

   T F The successor variety stemming algorithm is a language-dependent method.

   T **F** In relevance feedback, if all of the keywords are extracted from a relevant document to reformulate the query, the retrieval effectiveness will be better.

   T F Relevance feedback may find relevant documents that do NOT contain any of the keywords in the original queries.

   T **F** Google's Adwords program charges the advertisers based on the number of impressions of the ads.

   T **F** An important function of an enterprise search engine is to index and search data stored in databases (e.g., Oracle or MySQL databases).

   T **F** An important function of an enterprise search engine is to display relevant advertisements to the users.

   ~~T F Google's Adwords program charges the advertisers based on the number of impressions of the ads.~~

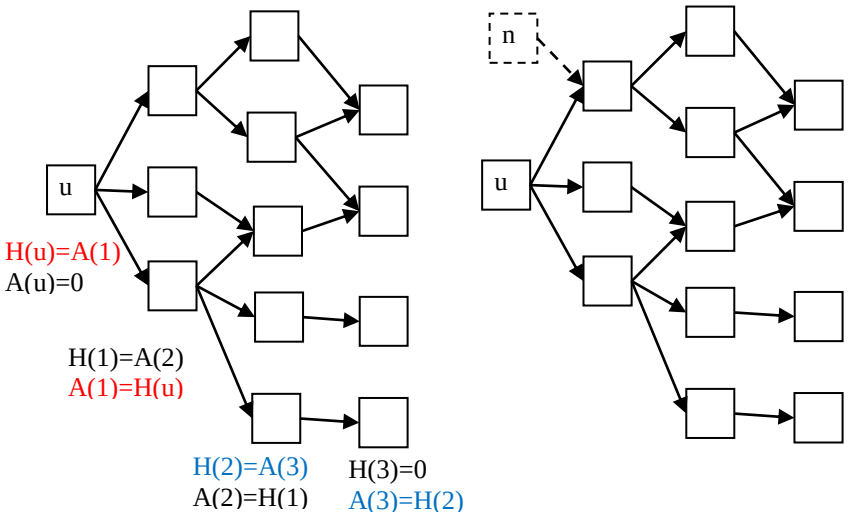
   T F When stemming has been applied to document terms, stemming must be applied to the query terms.

   T F When stopword removal has been applied to document terms, stopword removal must be applied to the query terms.

2. **[5]** Give three reasons for PageRank to become ineffective in enterprise search engine.

   (i) An enterprise website does not have many links
   (ii) Links do not imply authority
   (iii) Page that are generated from databases or in PDF/WORD formats do not have many links
   (iv) Links external to the enterprise website are not accessible
   (v) … …

3. **[10] (a)** Given a set of web pages linked as an acyclic graph as shown on the left below, discuss qualitatively the number of iterations it takes to arrive at (i) stable PageRank values and (ii) stable Hub and Authority values, given the initial weights are all equal to 1. Justify your answers. Note: you are not required to compute the exact PR, Hub and Authority values of the pages.



Note: an intuitive explanation is enough

PR of u does not change with iterations
1st level pages depends on u and become unchanged after 1 iteration
2nd level pages depends on 1st level and become unchanged after 2 iterations
3rd level pages depends on 2nd level and become unchanged after 3 iterations.

Hub / Authority: Takes many more iterations.
A(u) = 0
H(3rd level pages) <- 0
A(3rd level pages) <- H(2nd level pages) <- A(1st level pages) <- H(u)
H(u) -> A(1st level pages) -> H(2nd level pages) -> A(3rd level pages)
…

Since I mention "stable", it means that normalization is assumed, since without normalization the H and A values will keep on increasing. Again, it is better to compute the values for a few iterations to verify the behavior.

[10] (b) When a new page n is added to the web graph as shown on the right above, how does it impact the PageRank, Hub and Authority values?

The PR of n's child increases, which in turn increases the PR of its four children and grandchildren only.

Since it is a tree, the addition of n raises A(1st-level pages), which in turn raises H(u), which in turn raises A(1st level pages). The H and A values of the 2nd level pages are not dependent on A(1st level pages). Thus, the "ripples" stops at the 1st level pages.

Although the computation is not required, in order to assure the discussion is correct, it is better to compute the actual hub/authority values of all the pages for a few iteration to see the actual effect.

4. **[5] (a)** Given a block signature *s* of *F* bits, what is the optimal number of bits to set in *s*? Justify your answer. Discuss the effects when (i) only 1 bit is set in *s* and (ii) when all bits in *s* are set.

Half of F bits should be set to one. (i) many collisions, and (ii) no filtering power at all.

[7] (b) Given a hash function h() which outputs the following sequences of random integers when given different seeds:

h("data") = 197, 532, 26, 11, ...
h("info") = 178, 93, 376, 267, ...
h("text") = 11, 37, 87, 187, ...

A document has the block signature shown below with signature length *F* = 9 bits and each word setting 2 distinct bits in the signature. Which of the above words are potentially in the document? Give the steps that lead to your answer.

	0	1	2	3	4	5	6	7	8
block signature	0	1	0	1	0	0	0	1	1

data and info are potentially in the document but text is definitely not in the document.

Ans: For "data", 197%9=8, 532%9=1/ Thus, the word signature for "data" is

0	1	2	3	4	5	6	7	8
0	1	0	0	0	0	0	0	1

The same for "info" and "text" is:

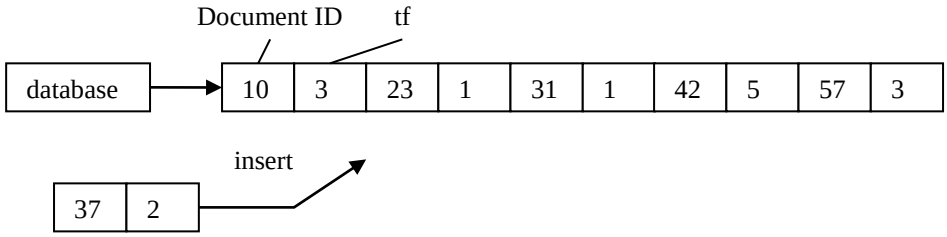
0	1	2	3	4	5	6	7	8
0 0	0 1	0 1	1 0	0 0	0 0	0 0	1 0	0 0

Comparing to the block signature, "data" and "info" are picked.

5. **[5]** (i) Describe <u>with an example</u> how "rank sink" is created in Google's PageRank method. (ii) What problem does a rank sink can potentially create? (iii) How does the PageRank formula resolve the rank sink problem?

(i) A rank sink is created when two pages point to each other and one of the page is linked to the rest of the web
(ii) PR values of the two pages increase indefinitely
(iii) The damping factor is introduced to suppress the PR values

6. **(a) [3]** Given the following postings list stored on disk, describe the steps to insert the postings item [37 | 2] in the posting list while maintaining the sorted order of Doc ID.. Further explain why this operation becomes more and more expensive as the postings list gets longer and longer.



[Assuming that the "database" postings list has already been found.] Read the postings list into main memory, identify the place at which the new postings item is inserted, i.e., after document 31. Insert the new postings item [37 | 2], and write the new postings back to the disk.

Note: Students may put in more details, e.g., to insert the new postings items, you may need to create a new empty postings list, copy everything from the beginning to the postings item form document 31, insert the new postings item [37 | 2], copy the rest of the postings list from the original list to the new list. (Then, write the new list back to the disk.)

What I am looking for is the read, insert new item into the right place to maintain the sorting order, then write the modified list back to the disk.

(b) [7] Discuss why the normalization factor (the denominator) in cosine similarity is very expensive to compute. Your discussion should explain the data needed and the amount of computation required for computing the normalization factor.

The document vector has to be computed, which incurs:
(i) Knowing all of the keywords in a document (not just the query terms)
(ii) Since a document typically contains a large number of keywords, computing the document vector length is time consuming.

7. **[10] (a)** Fill in the *precision*, *recall*, *fallout* and *NDCG* in the first five rows of the following table. It is assumed that there are a total of 100 documents, and there are only 4 relevant documents, which are marked with a √ in the first column. For NDCG, all irrelevant document score 0 and all relevant documents score 2.

	Rank	doc ID	Recall	Precision	Fallout Rate	NDCG
	1	--				
√	2	--				
√	3	--				
	4	--				
	5	--				
√	6	--				
...	. . .	...	...	...	...	
√	100	--				

[Do your calculation in the following space]

Ans:

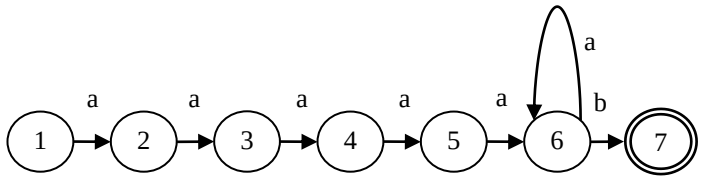
	Rank	doc ID	Recall	Precision	Fallout Rate	NDCG
	1	--	0	0	1/96	0
√	2	--	¼	½	1/96	0.5
√	3	--	2/4	2/3	1/96	0.62
	4	--	2/4	2/4	2/96	0.52
	5	--	2/4	2/5	3/96	0.52
√	6	--				
...	. . .	...	...	...	...	
√	100	--				

Detailed computation is omitted.

8. **[4] (a)** Using the **enhanced** KMP algorithm that prevents cascade mismatching, determine the number of character positions to shift in the shift array for the following pattern.

	1	2	3	4	5	6
Pattern	a	a	a	a	a	b
Shift (Basic)	1	1	1	1	1	1
Shift (Enhanced)	1	2	3	4	5	**1**

[4] (b) Draw a finite state automata for the pattern in (a). Discuss the performance of the FSA compared to the KMP and Enhanced KMP methods.



9. **[10]** A query returns the following ranked documents: d1, d2, d3, d4, d5, d6 and the user clicks on d1 and d4 (if order of the clicks is needed, assume d1 is clicked first). What are his preferences for:
(i) Click > Skip Above
d4>d2, d4>d3

(ii) Last Click > Skip Above

d4>d2, d4>d3

(iii) Click > Earlier Click

d4>d1

(iv) Last Click > Skip Previous

d4>d3

(v) Click > No-Click Next

d1>d2, d4>d5

10. **[5] (a)** This question requires you to know the specifics of the term project requirements.

T / F The project requires you to crawl a web site in breadth-first order.

T / **F** The project requires you to keep the parent-child relationship between two pages for implementing PageRank.

T / F The project requires you to implement the cosine similarity function.

T / **F** The project requires you to implement the peak-and-plateau stemming method to stem the keywords before inserting them into the index.

T / **F** The project requires you to use cookies to remember the users of your search engine.

[5] (b) Write a crawling function which uses breadth first search to crawl a specific number of web pages given a starting URL. Note: The function behaves as in the following. You should fill the function body.

```
public Vector<String> crawling(String startURL, int numPagesToBeCrawled) {}
```

- Only unique URLs should be output, which means you need to filter out the redundant URLs (two strings of URL are exactly the same).
- Assume there are abundant URLs to be crawled, which means the number of web pages in the website is much larger than the "numPagesToBeCrawled"
- Java language is preferred (other programming language is also OK). You can use any class provided by Java. Necessary comments should be added.
- The syntax does not need to be exactly correct, but the general flow of the program should be precise and clear.

```
public Vector<String> crawling(String startURL, int numPagesToBeCrawled) {
                //All the links extracted from the startlink;
    Queue<String> linkQueue = new LinkedList<String>();
        //Hashmap to store all the unique URL page;
    Vector<String> page_list = new Vector<String>();
        int remain = numPagesToBeCrawled;
                // Insert start url in Queue
    linkQueue.offer(startURL);
        //Take turn to fetch each link stored in quere;
    while(linkQueue.peek()!=null && remain!=0){
        String now_url = linkQueue.poll();  //Retrieve and remove
                        // Extract all child links
        Vector<String> child_url_list = extractLink(now_url);
                        // Push the children URL to queue.
        for(String eachURL: child_url_list){
            linkQueue.offer(eachURL);
            //System.out.println(eachURL);
                }
                 if (!page_list.contains(now_url) ){
                        page_list.add(now_url_key,now_url);
            remain = remain - 1;
                 }
        }
        return page_list;
}
```