Search Engine

WS 09/10

Exercise Sheet 1

By: **Waleed** butt

University of Freiburg

Dated: 26-Oct-2009

Exercise 1

User Account and Introduction is done on wiki. My Username = Waleed

Exercise 2

I downloaded around 1500 RFC Documents from http://www.rfc-editor.org

Exercise 3

Parser or Token Generator is implemented in C# using Microsoft .Net 3.5 Framework. See attached file. There is two phases of the Parser.

Tokenization

a. Generates the Tokens (shown in slides) in raw form.

```
foreach (string filename in FilesName)
                {
                    //string fullpath = FilesPath + "\\" + filename;

                    rdr = new StreamReader(FilesPath + "\\" + filename);

                    strArr = Linebuffer.Trim().Split(' ');

                    foreach (string token in strArr)
                    {
                        if (token != "")
                        {
                            Tokens.Add(new KeyValuePair<string,
string>(token, filename));
                        }
                    }
                }
```

Inverted Index List Generation

b. Create Inverted index list from Token List.

```
 //Generate Inverted Inex

SortedList<string, List<string>> InvertedIndex = new SortedList<string,
List<string>>();

  foreach (KeyValuePair<string, string> item in Tokens)
            {
if (InvertedIndex.Keys.IndexOf(item.Key) >= 0)
 {

  (InvertedIndex[item.Key]).Add(item.Value);
 }
  else
   {
    List<string> templist = new List<string>();
    templist.Add(item.Value);
```

```
InvertedIndex.Add(item.Key,templist);        }     }
```

Exercise 4

There were Millions of the Two-Words Query for Exact one record with huge time and lost of nasy charactors in it. I did filter out some of them for sampling. For Reference you can see file in zip with name ***TwoWordsQuery.tok***

```
authenticated: : authentication)
authenticated: : automatically
authenticated: : badly
authenticated: : based
authenticated: : before
authenticated: : beginning
authenticated: : behalf
authenticated: : behalf,
authenticated: : believe
authenticated: : bend
authenticated: : beyond
authenticated: : bill.
authenticated: : billed
authenticated: : both
authenticated: : broad
authenticated: : building
authenticated: : built
authenticated: : buried
authenticated: : busy
authenticated: : But
authenticated: : cannot
authenticated: : case
authenticated: : case,
authenticated: : cases
authenticated: : cause
authenticated: : caused
authenticated: : change
authenticated: : charge.
authenticated: : charged)
authenticated: : checked
authenticated: : choose
authenticated: : chose
authenticated: : clearer
authenticated: : clearly
authenticated: : code
authenticated: : code,
authenticated: : come
authenticated: : coming
authenticated: : commands)
authenticated: : Comments:
authenticated: : communication
authenticated: : concerned
authenticated: : conclusion.
authenticated: : conflict
authenticated: : confusion
authenticated: : connects
authenticated: : consider
authenticated: : contribute
authenticated: : correct
authenticated: : correspondence
```

```
authenticated: : correspondingly
authenticated: : course
authenticated: : course,
authenticated: : criticism
authenticated: : cumulative
authenticated: : customers
authenticated: : daemon
authenticated: : daemon"
authenticated: : damage
authenticated: : decided
authenticated: : deep
authenticated: : deliberately
authenticated: : design
authenticated: : desirable
authenticated: : desires:
authenticated: : determine
authenticated: : determining
authenticated: : develop
authenticated: : development.
authenticated: : different
authenticated: : different.
authenticated: : difficult
authenticated: : directories
authenticated: : directory?
authenticated: : discussions
authenticated: : disk
authenticated: : distinct
authenticated: : distinguishing
authenticated: : do
authenticated: : don't
authenticated: : each
authenticated: : earlier
authenticated: : efforts
authenticated: : Either
authenticated: : enable
authenticated: : end
authenticated: : END
authenticated: : enough
authenticated: : entry
authenticated: : established.
authenticated: : etc.
authenticated: : evaluate
authenticated: : examine
authenticated: : examines
authenticated: : execute
authenticated: : executing
authenticated: : exist
authenticated: : exist,
authenticated: : existing
authenticated: : expands
authenticated: : explained
authenticated: : explicitly
authenticated: : fairly
authenticated: : familiar
authenticated: : far
authenticated: : fewer
authenticated: : file.
```

```
authenticated: : files,
```

Exercise 5

Complexity

Just because of short time I was not able to work on this side. However I tried to use optimal approach for this but the algorithm is still in need some enhancements domain. This approach is bit close to brute force with two optimal points.

1) For Finding match between two Words in Inverted list it run less than $O(n^2)$ or close to O(nLogK).
2) We have variable size of linklist attached with each word.

My approach could be far better than if I use ID for each file rather than using filename as string.

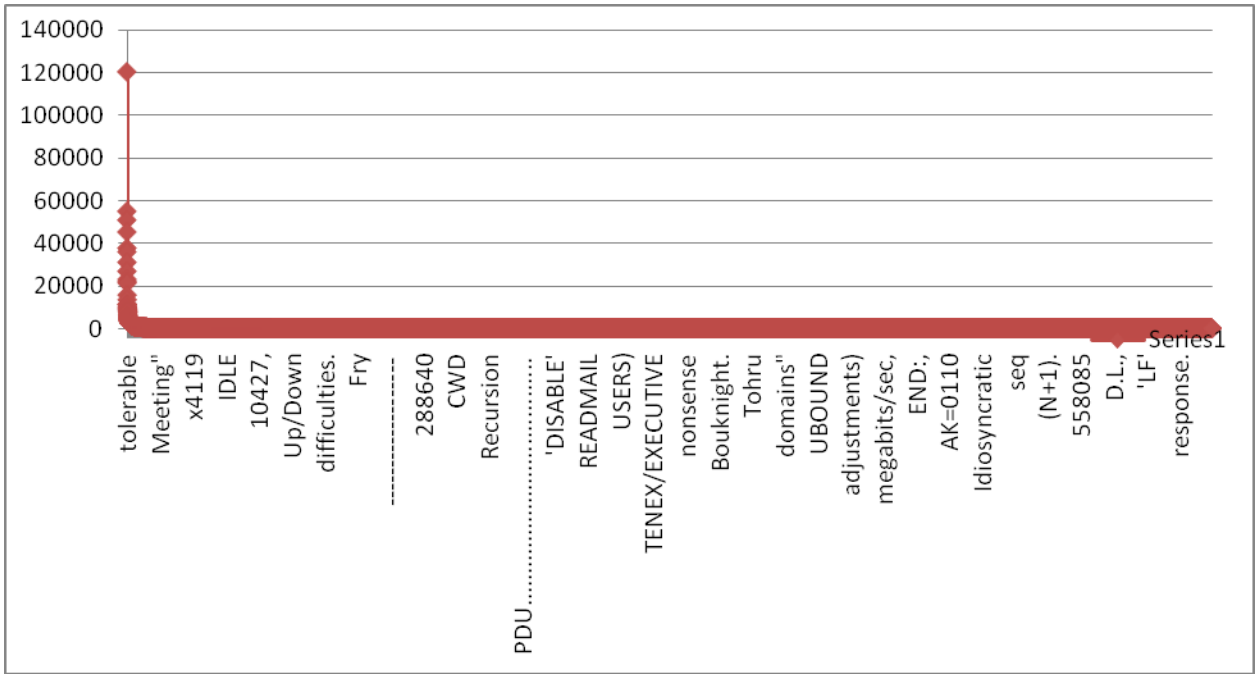
Frequency Computation

```
List<KeyValuePair<string, int>> Freq = new List<KeyValuePair<string, int>>();

foreach (KeyValuePair<string,List<string>> item in InvertedIndex)
  {
   Freq.Add(new KeyValuePair<string,int>(item.Key,item.Value.Count()));
   }
```

See also: Excel file in Zip

Graph



See also: Excel file in Zip

Exercise 6

Source Code with Supported Documents is placed on Wiki.