Information Retrieval Assignment

Teaching Objective:

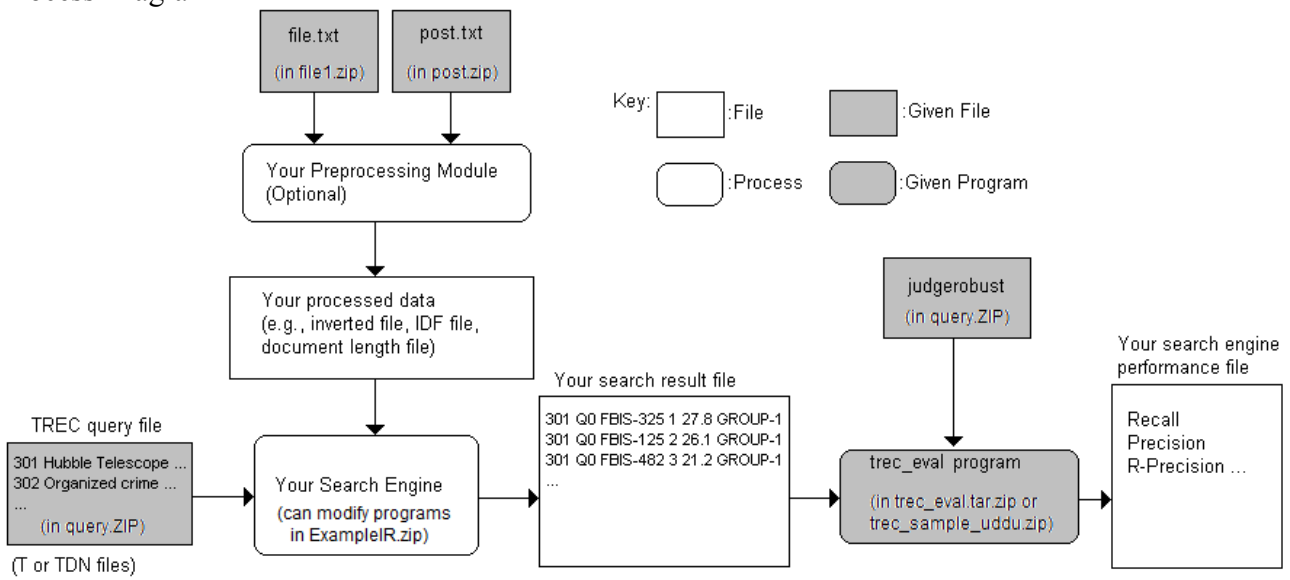
1. Gain hands-on experience in developing a search engine (objective ii and v);
2. Gain hands-on experience in the evaluation of search engines (objective ii and v);
3. Practice communication skill (Attributes of all-roundedness 3 and 4) during demo;

The Problem: You have joined an information retrieval evaluation workshop hosted by me. You need to form a group of size that is between 4 and 6 (inclusive). You should hand in details (i.e., full name, student id, group leader and his/her mobile phone contact) to me by **October 25th, 2012**. This workshop seeks to find the most effective and most efficient system that you can design and implement. You are given a set of TREC queries, a set of TREC relevance judgments and some inverted index data. You are required to further develop the system using MS Access or develop another system in C++ (in part or in whole) or another system developed in Java (in part or in whole) to retrieve and rank the given set of documents for the given set of queries. Due to the distribution agreement signed by me with TREC, I cannot distribute the TREC documents to you. However, the inverted index data is available for you to build a search engine.

Given Material (available in WebCT):

1. irproj.zip (programs, readme file, etc.)
2. file.txt (contains the mapping between file identifier and document name used for TREC evaluation)
3. post.txt (contains the inverted index information of the collection)
4. trec_eval.tar,gz (TREC evaluation program and its Windows version)
5. query.zip (Query and relevance judgement files)
    a. judgerobust: contains the relevance judgment of each query
    b. queryTDN: long queries
    c. queryT: short queries

Process Diagram



TREC query format (queryT and queryTDN in query.zip) of each line as follows:

XXXΔ<Query>

where **XXX** are three digits that indicate the query number, **Δ** is the space character and **<Query>** is the content of the query. Figure 1 is an example of a set of queries.

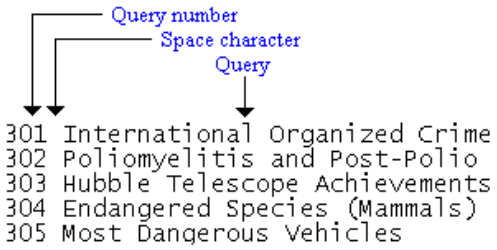


Figure 1: Example TREC queries

TREC Search Result Format (see readme in trec_eval.tar.gz)

The format of the search result (juderobust in query.zip) per line is as follows:

XXXΔQ0Δ<DOC-ID>ΔRΔSΔ<RUN-ID>

where:

XXX are the three digits to specify the query number;
Δ is the space character;
'Q0' is a constant string;
<DOC-ID> is the document identifier;
R is the rank number of the document (1 to 1000 for each query);
S is the similarity score (in descending order);
<RUN-ID> is a string that specifies the run identifier.

Example:

```
301 Q0 FBIS-325 1 27.8 HKPU-1
301 Q0 FBIS-178 2 23.1 HKPU-1
…
```

means:

query number = 301
document identifier = FBIS-325
rank = 1
similarity score = 27.8
run identifier = HKPU-1

Deliverables:

1. Technical Report: 11 points, 1.5 spacing, 20-50 pages. The report should give details of your search engine (e.g. architecture, retrieval models, and ranking techniques), experiments (e.g., experimental design, result and analysis, and conclusion) that you have done, etc. The experiments should report at least the recall-precision curves, the R-precision and the top 10 document precision. When you report the time-efficiency, you should record the speed using the indicated machine in the COMP PC lab. You should also specify the RAM size of the PC and the processor type plus the speed of the system.
2. Search results: Your results should be produced in the TREC search result format (see below) and these results should be stored in the file T.ret and TDN.ret for queryT and queryTDN, respectively. Each query should return at most the top 1000 retrieved documents.
3. CD: It contains your program and relevant data so that I can run your system in one of the COMP PC lab for verification. You should indicate in which COMP PC lab your system is able to run (preferably with an indication of which machine). If I cannot run your program in the indicated COMP PC lab (and machine), marks will be deducted.

Important Dates:

1. Submit your technical report (hardcopy) and CD/DVD (contains your report, powerpoint, search results, etc.) by **December 20th, 2011** to the teaching assistant (Contact details are shown in point 2) just before the presentation. Late submission will result in mark deduction.
2. You should arrange with the teaching assistant (Sam Sin, Tel: 2766 7130, Email: sam.sin@polyu.edu.hk) to demo your system between **December 27th 2012 and January 6th, 2013**. Failure to do so will result in mark deduction. The teaching assistant will run your program using his/her data. If your program does not work, substantial amount of marks will be deducted because we cannot evaluate whether your system is working or not.

Grading

1. If you produce a **basic working system**, then a grade C/C+ will be awarded. You have demonstrated basic understanding of the subject. A basic working system includes the ability to:
    a. automatically **read in a TREC query file** when a query file is specified by the user and;
    b. automatically **produce the search results in TREC format** based on some taught retrieval model so that the trec_eval program can be readily used to obtain the performance of the system. If the search result returned 0.0 for precision, recall, or mean average precision, then a grade lower than C might be awarded because it does not show that your system is working.
2. For grade B/B+, you are required to at least achieve the basic working system (i.e., the level of C/C++ grade). In addition, if you are able to implement more than one retrieval model, you will need to **compare and analyze** the merits of one retrieval model over the other. The analysis needs to be substantiated by **measurements** to demonstrate that your analysis is correct. Then, you can be awarded a grade B/B+. Alternatively, if you have only a single retrieval model, you can **analyze the problems** of your retrieval system and **improve** your retrieval system with a new/enhanced method. Then, you can analyze whether the

new/enhance method is performing better than the original method. Again, the analysis needs **measurements** to substantiate the new/enhanced method, and the improvement of the retrieval system needs to be **measured** as well.

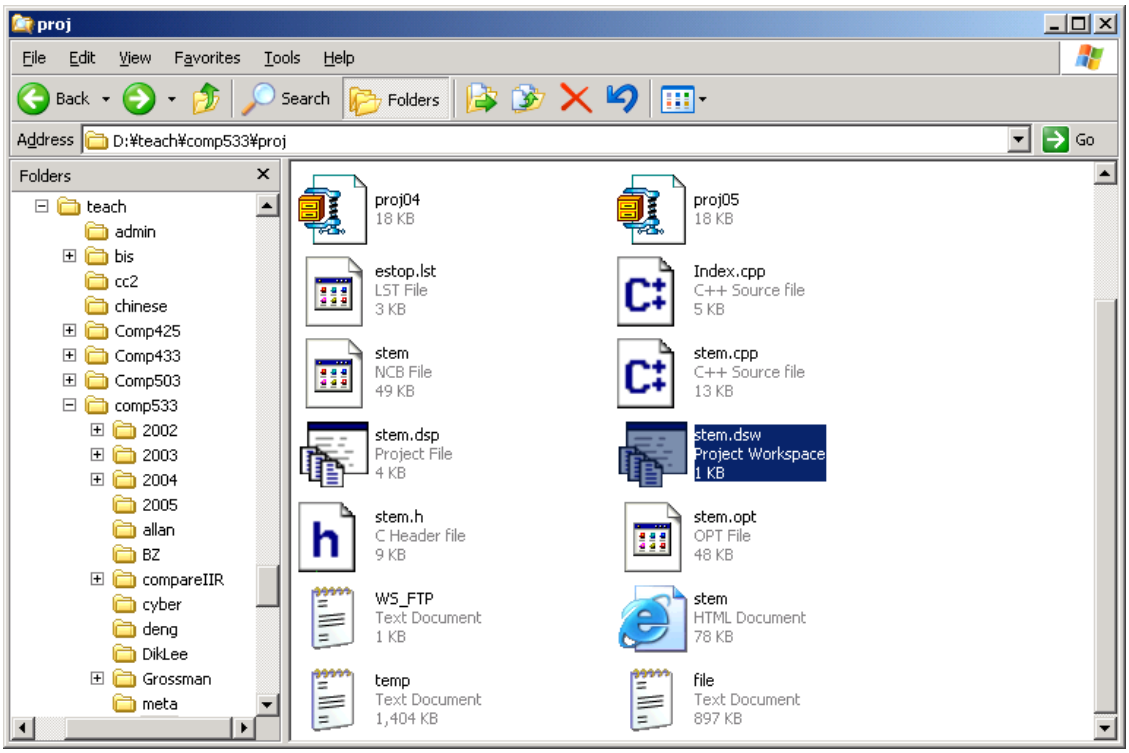
3. For grade A/A+, you are required to at least achieve the level of grade B/B+. In addition, you need to have *some innovative elements* in the assignment. The **innovation** needs to be substantial in order for the project to be avoided grade A/A+. Hence, trivial innovations do not count. Example of substantial innovation includes inventing a new retrieval model that can substantially improve the retrieval effectiveness of your basic system or improved system. Again, you need to substantiate that your innovation is effective by gathering **evidence**.
4. **You can use the programs provided to you by this subject for this assignment, but you are <u>not allowed</u> to use any open source IR programs**. You cannot share programs with other groups unless these programs are provided by us. Fail grades will be awarded to those who are found to participate in plagiarism or in copying other group's programs.

Initial Set Up Procedure:

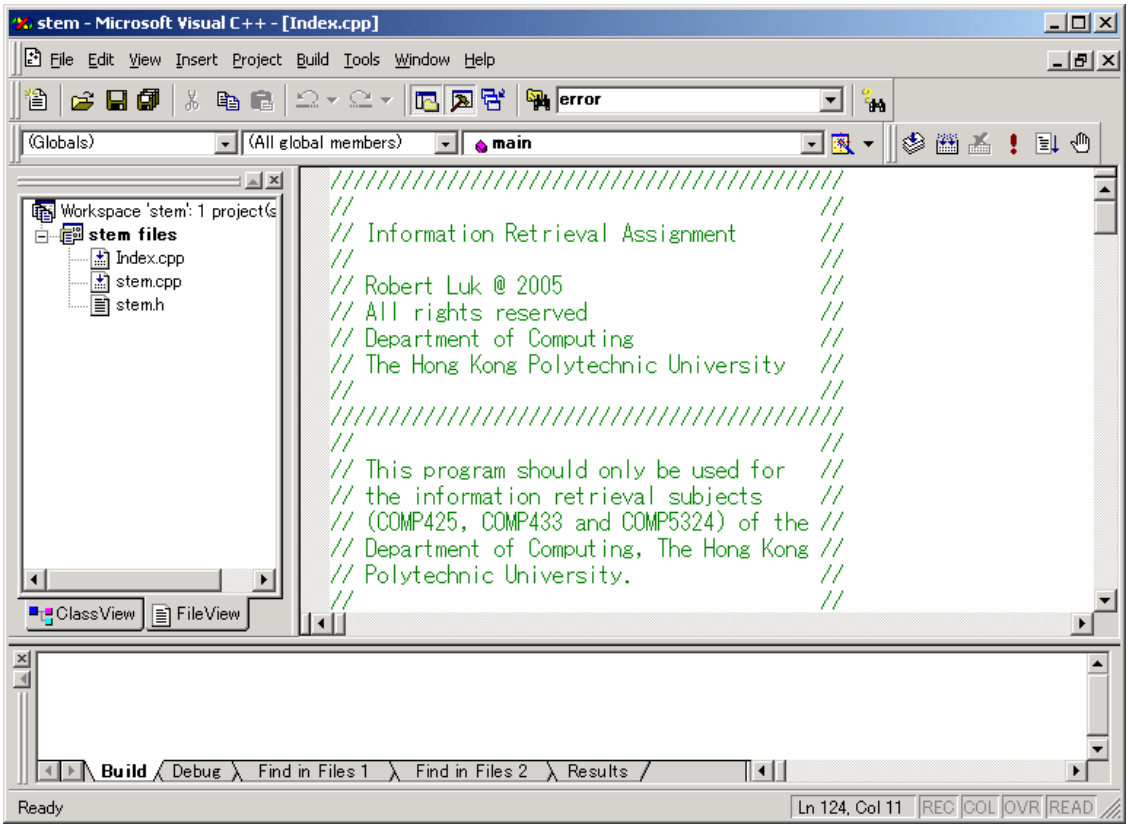
The first 10 steps are for reference only, because we can not release our TREC documents to you. The processing is already done for you, but it shows how the data are obtained so that you have a complete picture of the process.

From step 11, you can follow it and use MS Access to build a search engine (although not recommended because it is too slow) or use it to preprocess data.

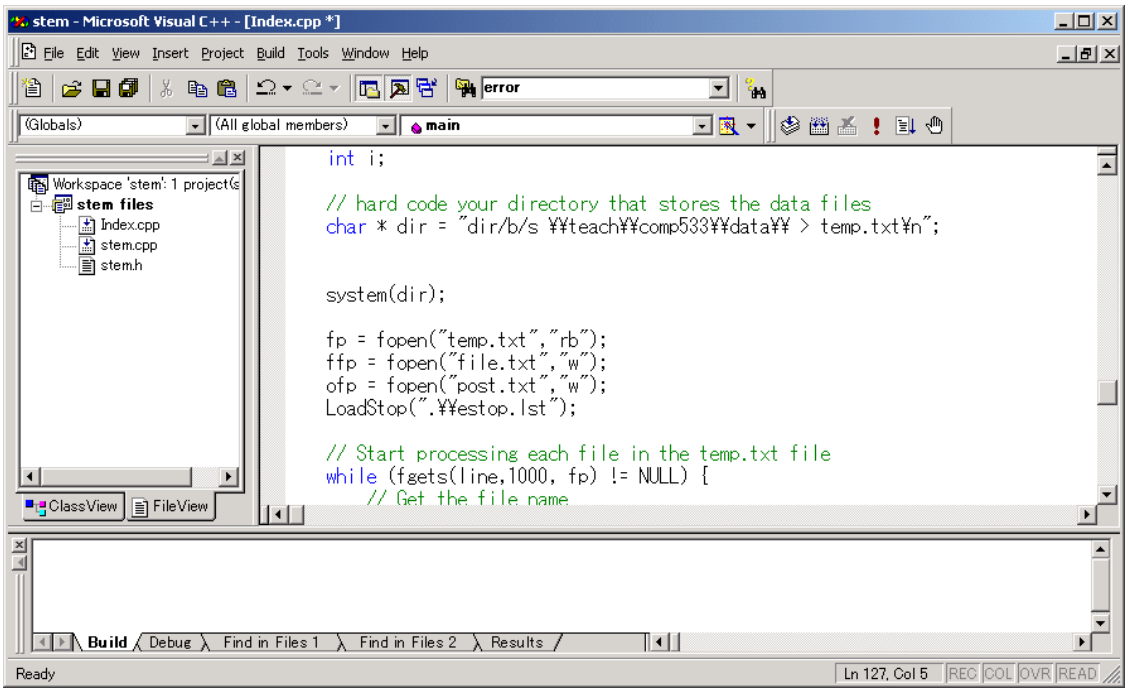
Step 1. Download the IRProject file (For illustration) [done]
Step 2. Unzip the IRProject file (For illustration) [done]
Step 3. In MS Windows with Visual C++ installed, click on **stem.dsw** (which is in reverse video in the following figure) (For illustration) [done]



Step 4: You should see MS Visual C++ development environment as follows: (For illustration) [done]



Step 5: Scroll down the **Index.cpp** file until you see the **dir** variable declared as follows: (For illustration) [done]



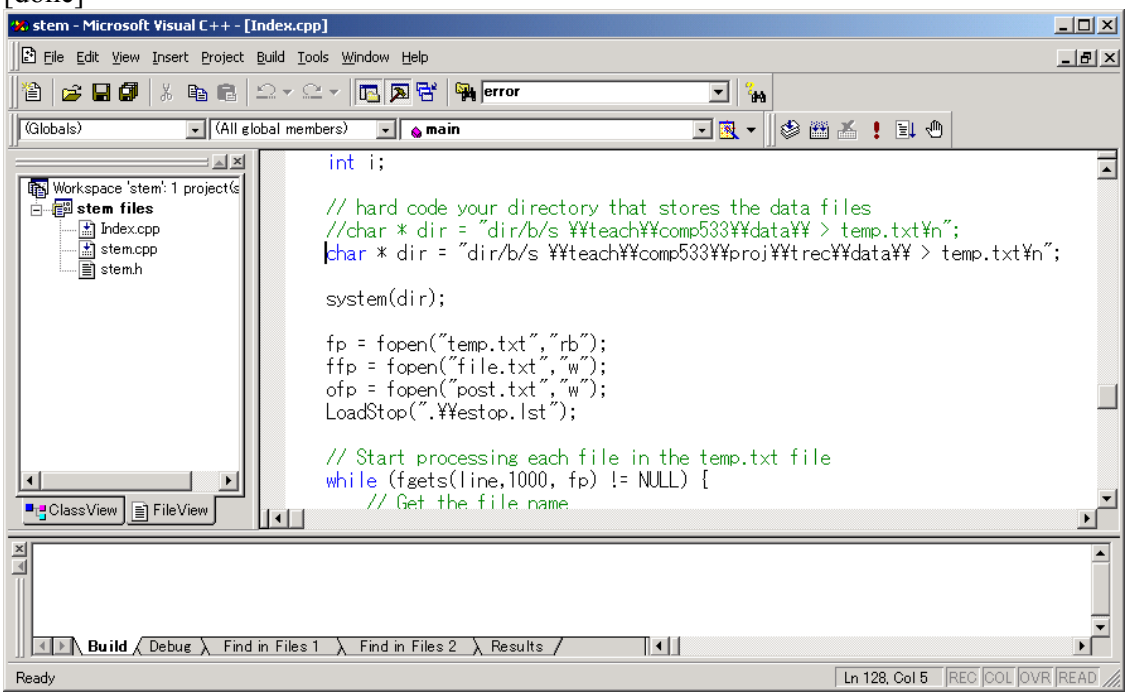
Step 6: Change the directory specification of the **dir** variable to the directory that stores your data. (For illustration) For example, we have changed the directory to:

```
\\teach\\comp533\\proj\\trec\\data\\
```

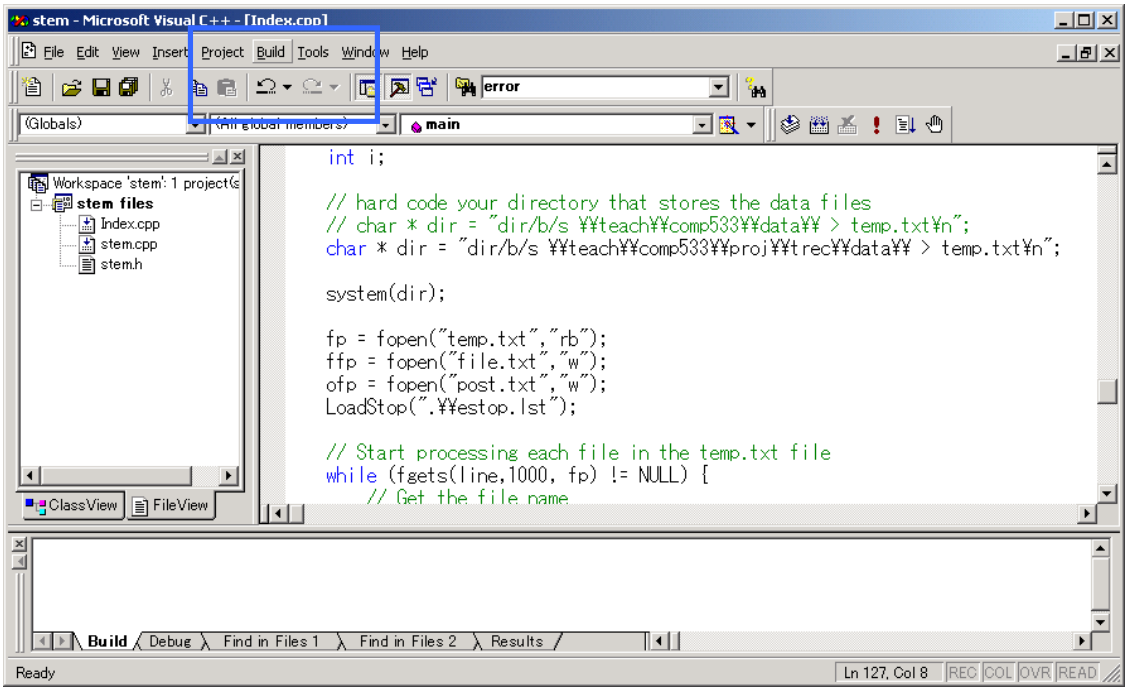
and we have specify the value of the **dir** variable as follows:

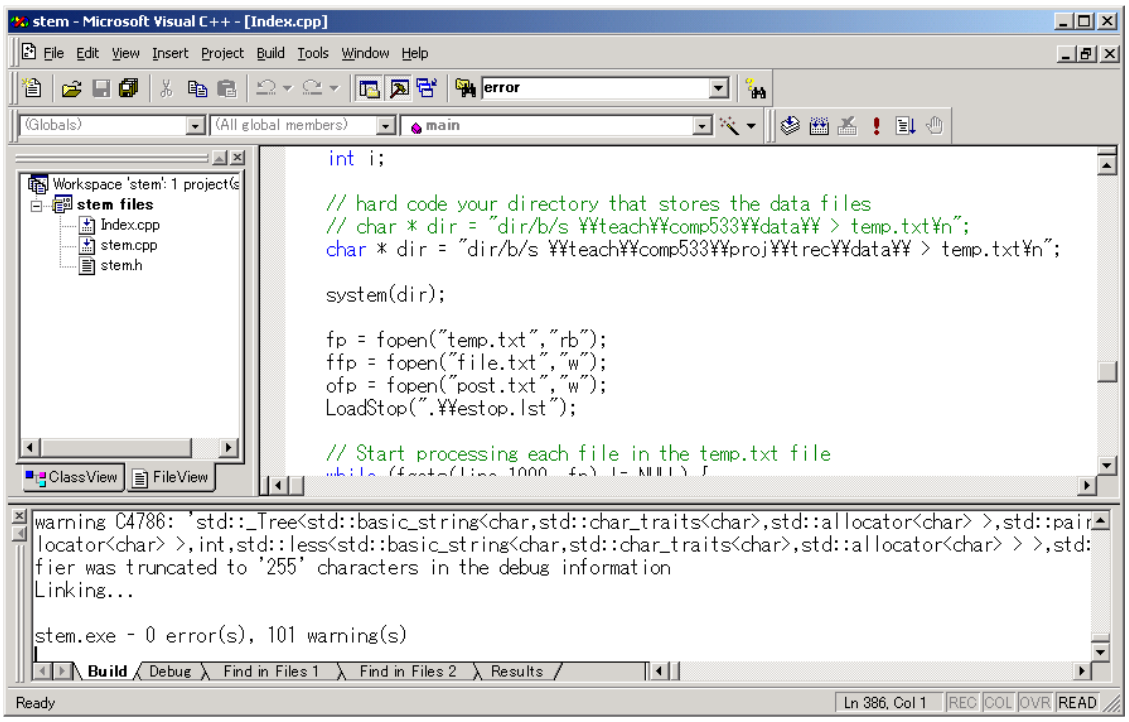
```
char * dir = "dir/b/s \\teach\\comp533\\proj\\trec\\data\\ > temp.txt\n";
```

[done]

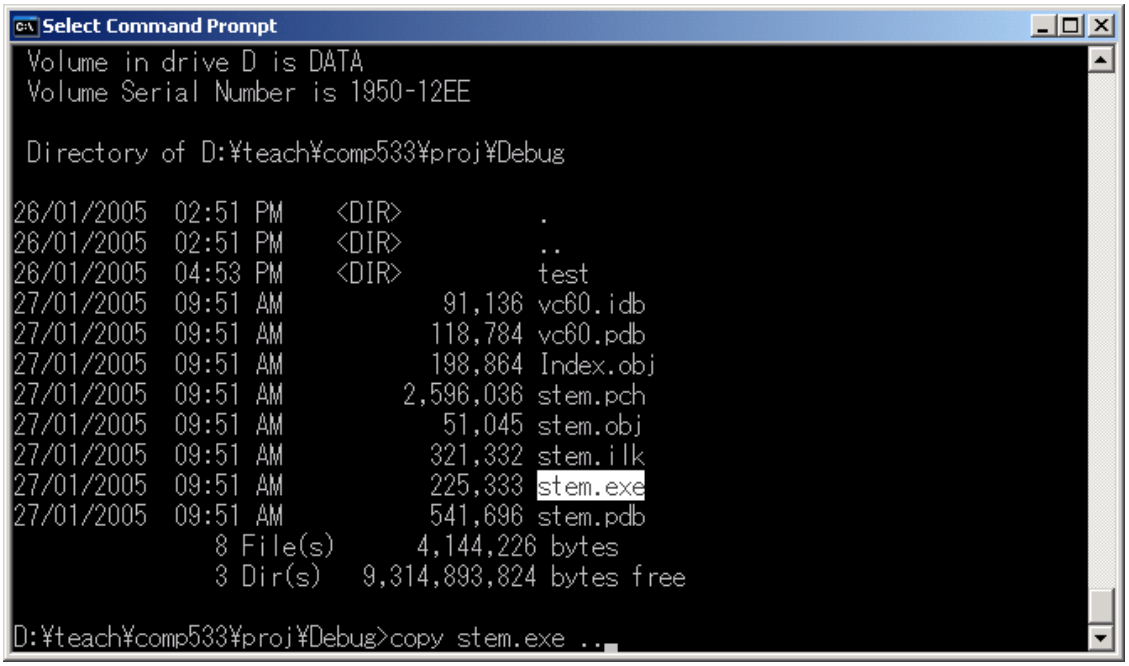


Step 7: Compile the program. Click the **Build** menu (top) and click the **Rebuild All** option in the pull down menu. (For illustration) [done]

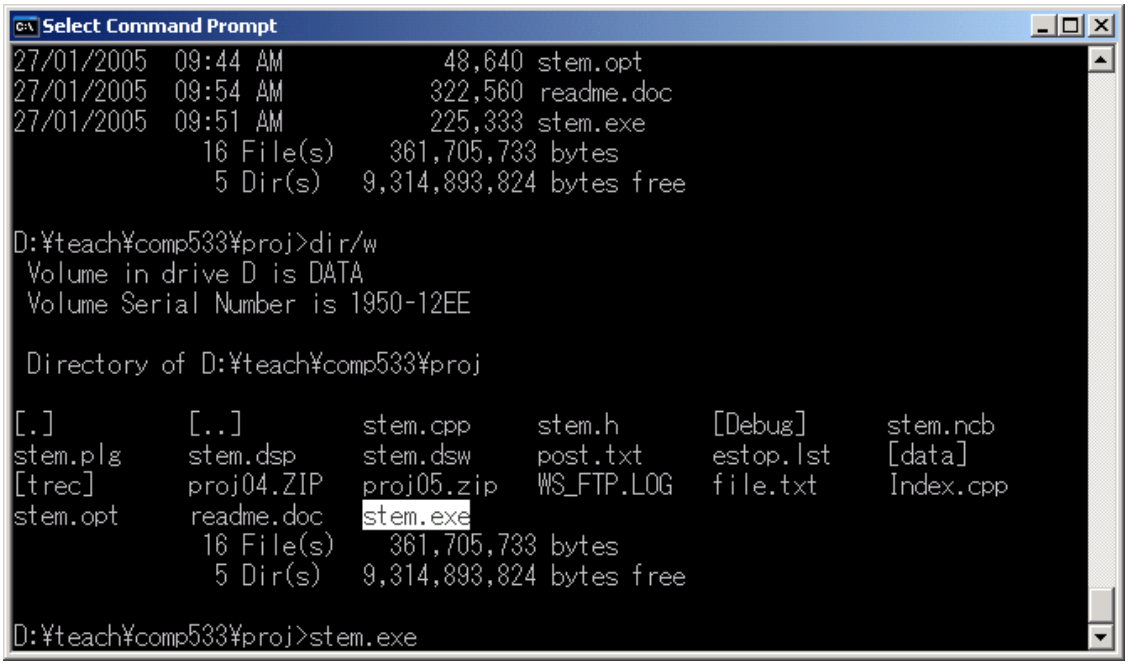




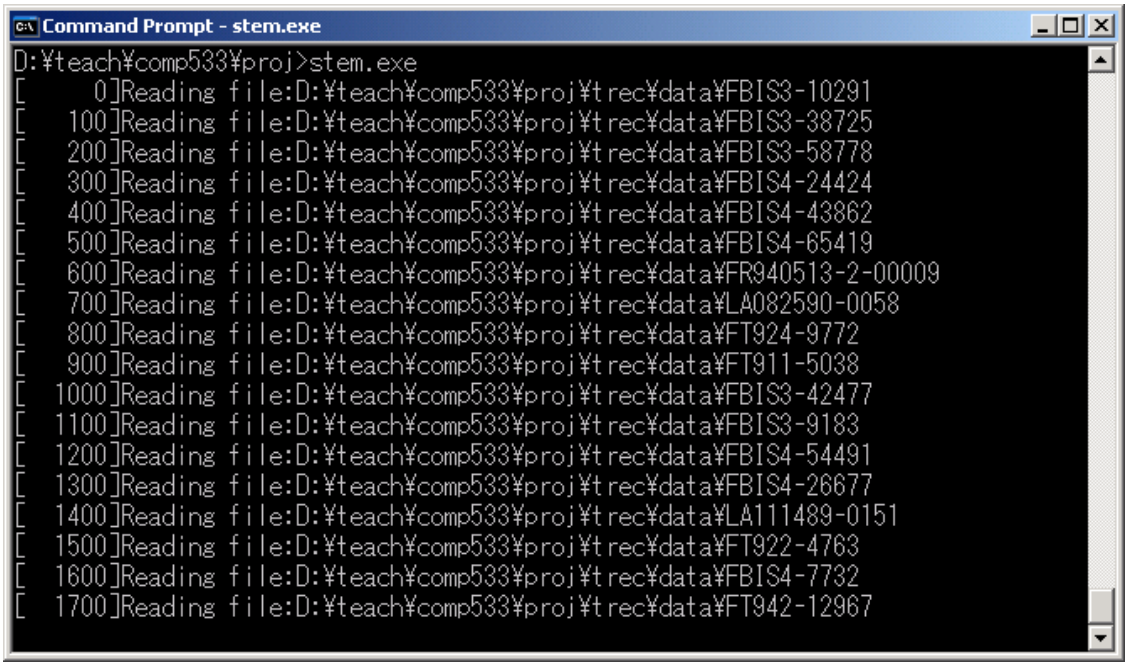
Step 8: After compiling the program, copy the compiled program **stem.exe** (which is stored in the child directory **Debug** of the current directory) to the current directory. (For illustration) [done]



After copying:



Step 9: After making sure that the current directory has the **estop.lst** file (i.e., the stop word list), you can run the program **stem.exe** as above. When **stem.exe** is running, it will report it has processed every 100 documents as in the screen shot below. (For illustration) [done]



Step 10: The program, stem.exe, produces two files: (For illustration)

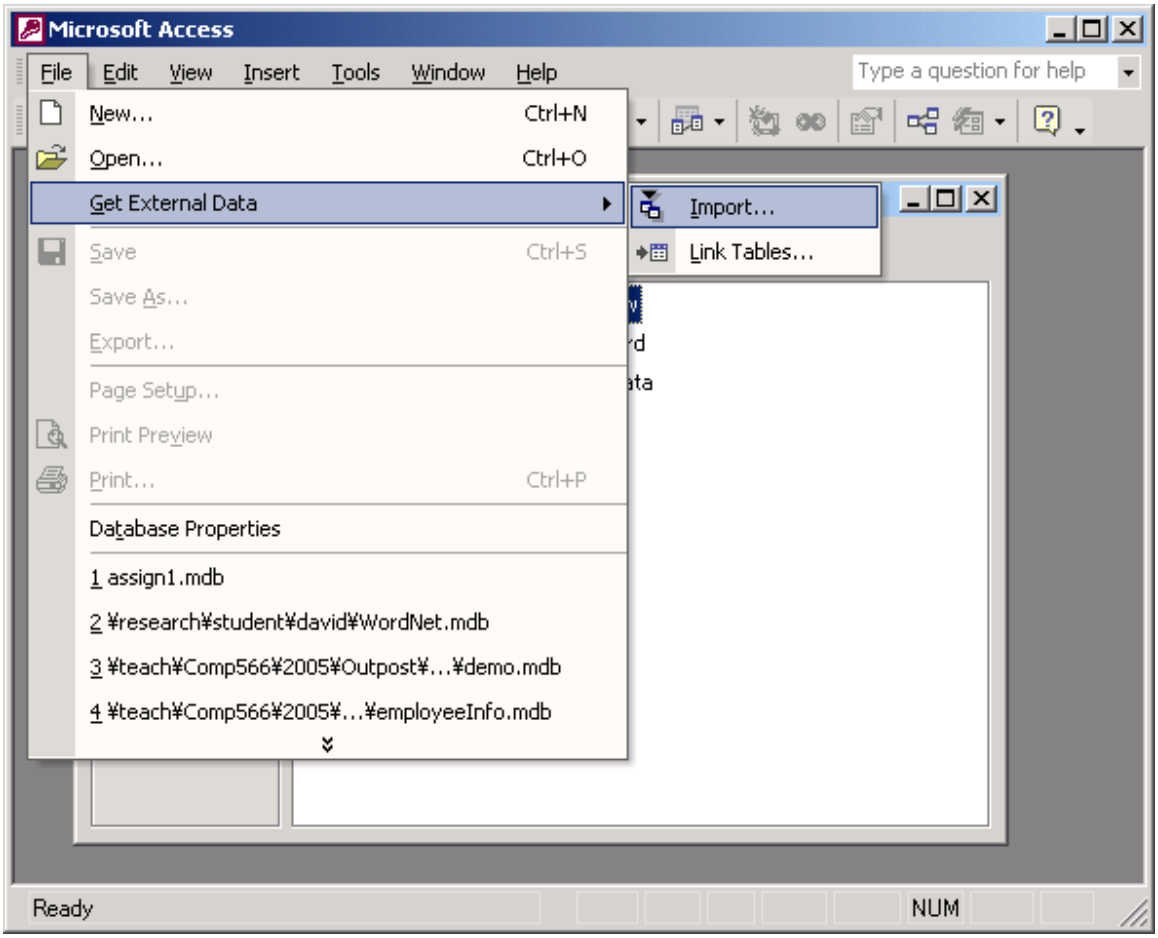
post.txt: this file contains the posting data which has the following format

```
field name    term          file-id       logical word position
example       chinesE       2             13
```

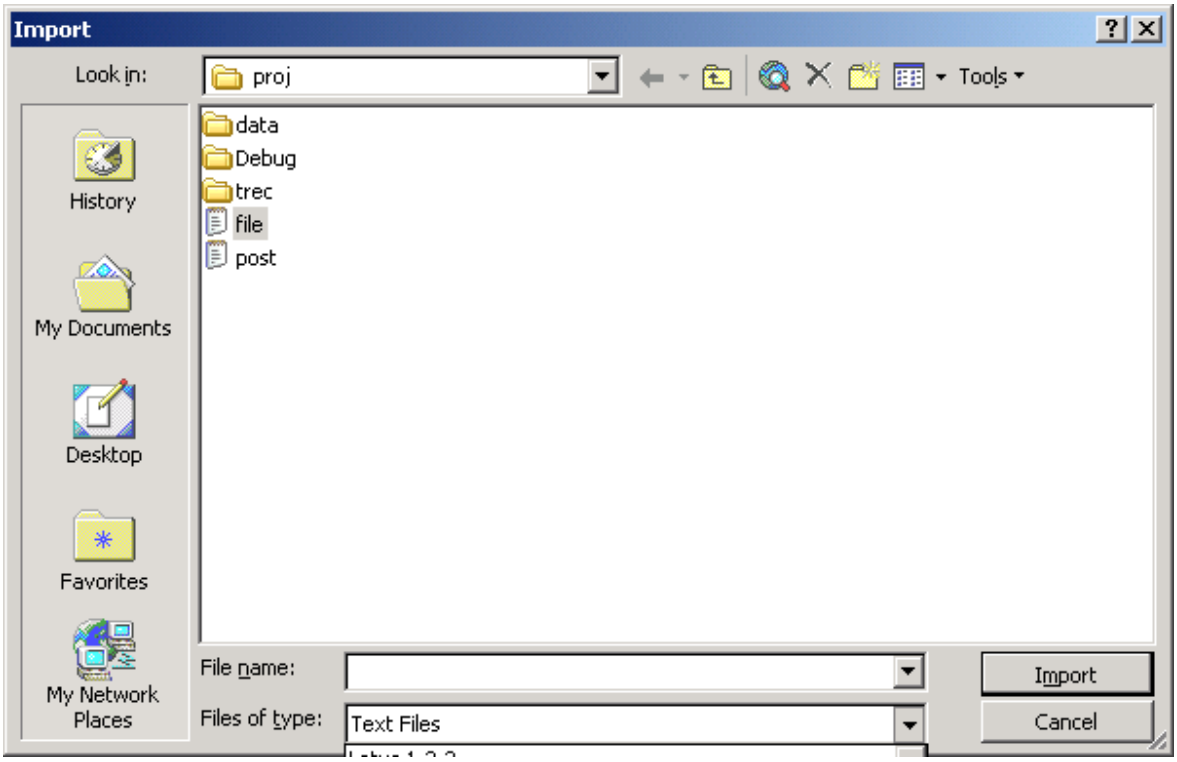
file.txt: this file contains the file id, name and path data. The format is as follows:

```
field name    fileid doclen status docid         path
example       13     377    @      FBIS3-14196   D:\teach\comp533\proj\trec\data\FBIS3-14196
[done]
```

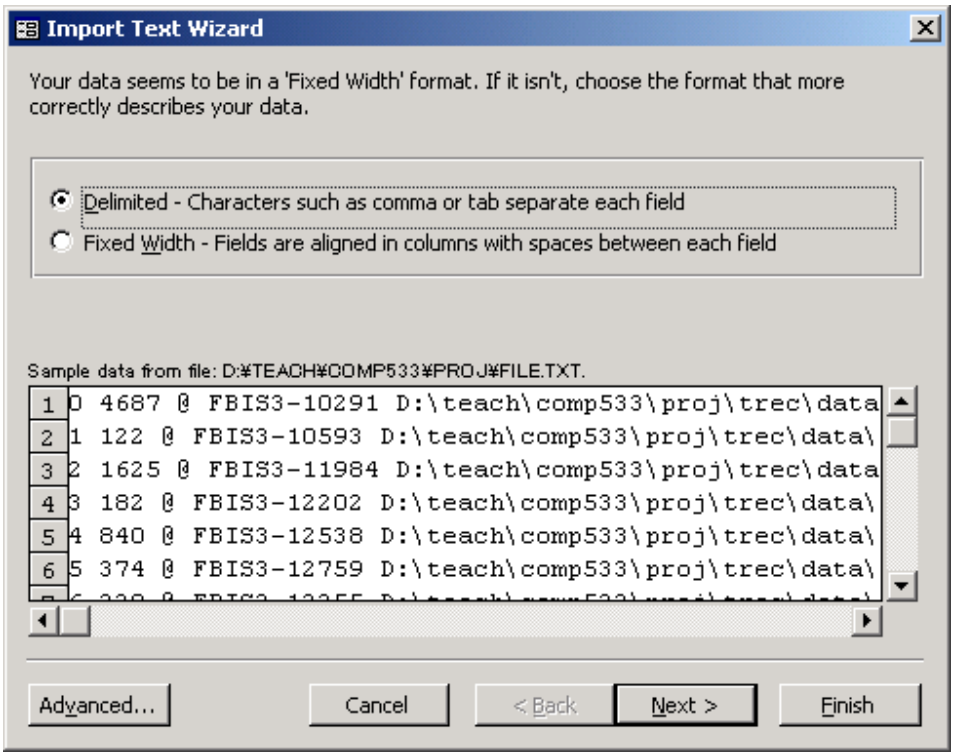
Step 11: Import these two files into your new MS Access database into two tables: post table for data in post.txt and file table for data in file.txt



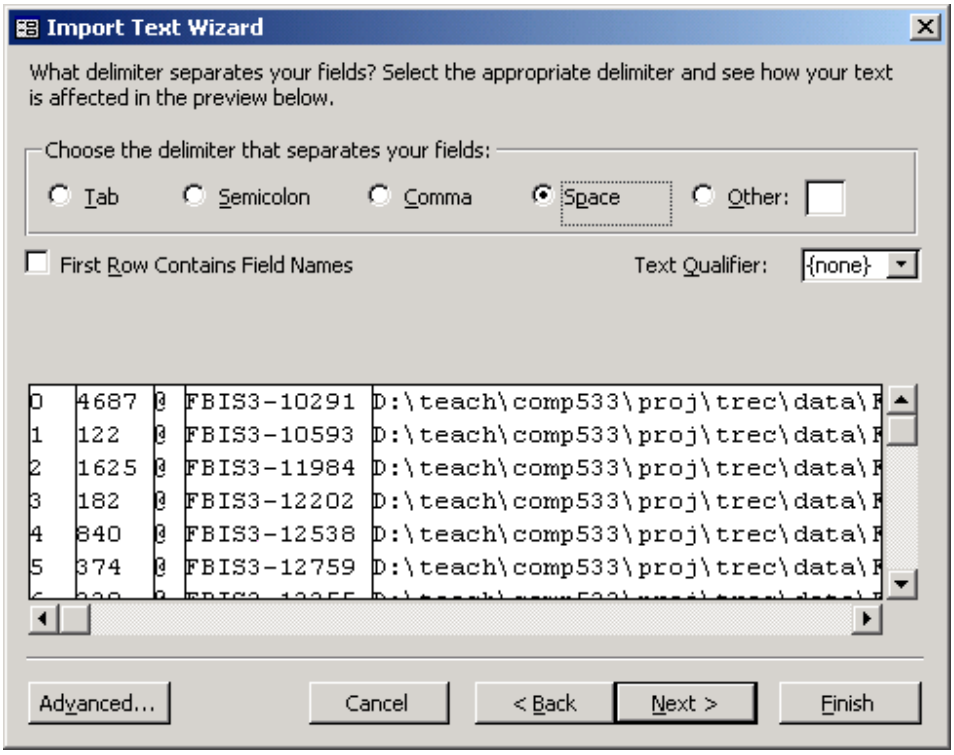
Import based on text files:



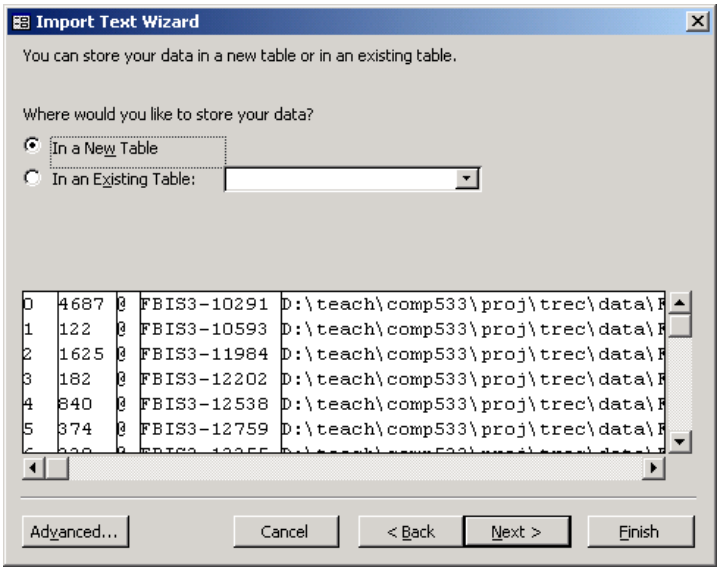
Step 12: Click on **Delimited** and then **Next**.



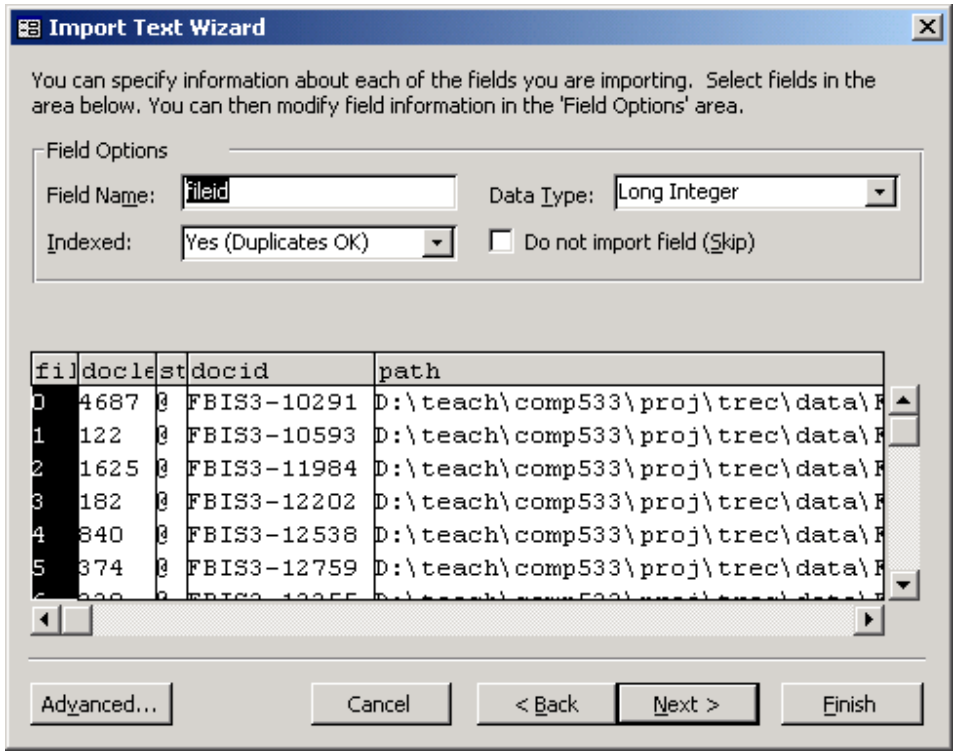
Step 13: Click the **Space** radial button and press **Next**.



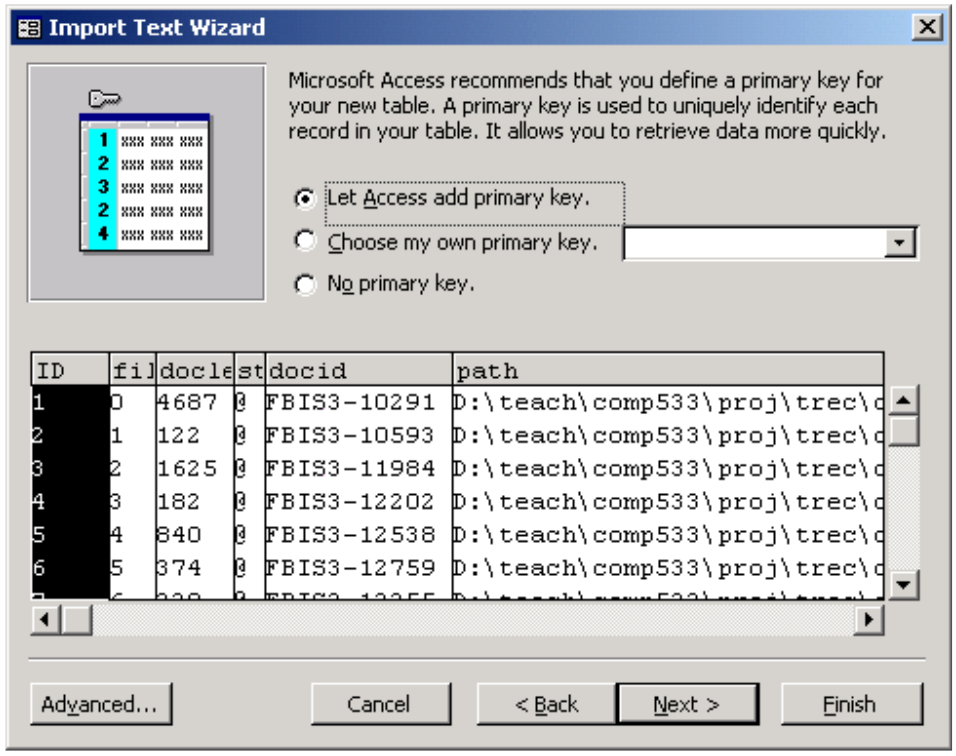
Step 14: Put the data into the new table (file.txt) and press **Next**.



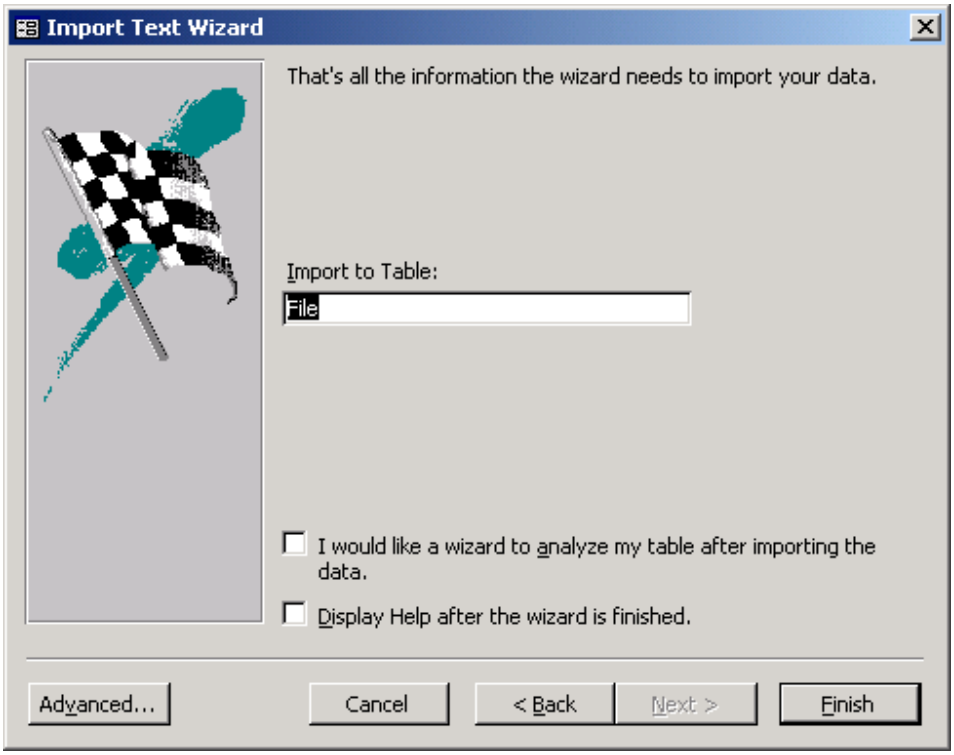
Step 15: Enter the names of each file.



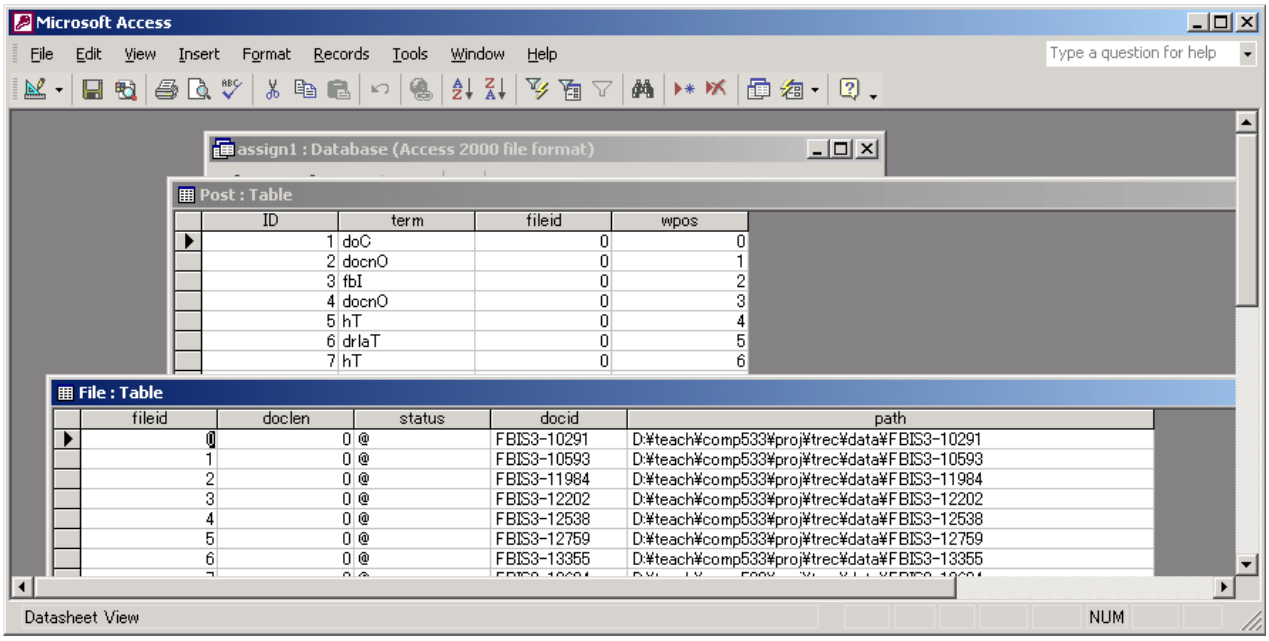
Step 16: Set the primary key to **fileid**.



Step 17: Enter the name of the able as **File**.



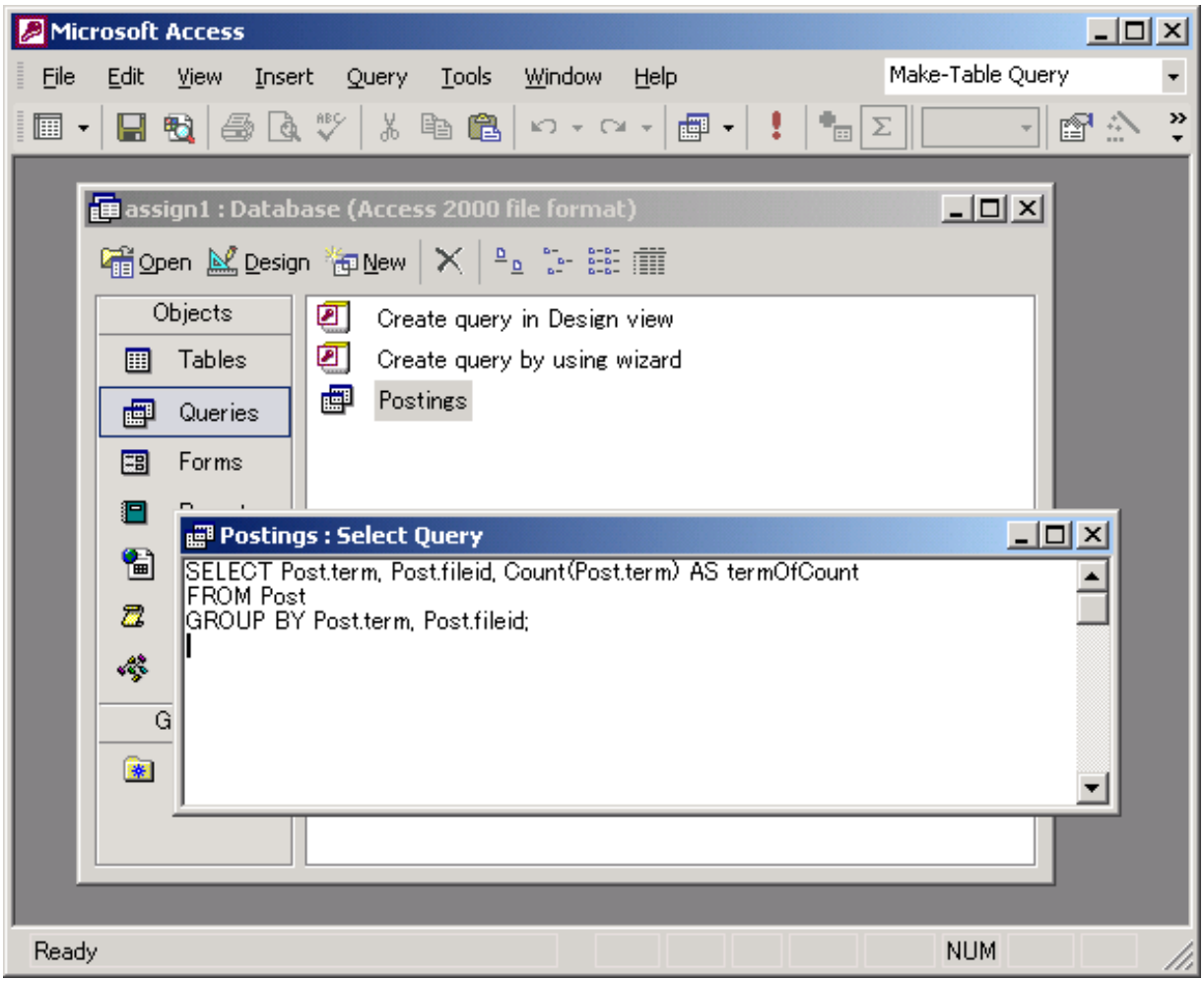
Step 19: After importing the post.txt and file.txt, your database should look like the following:



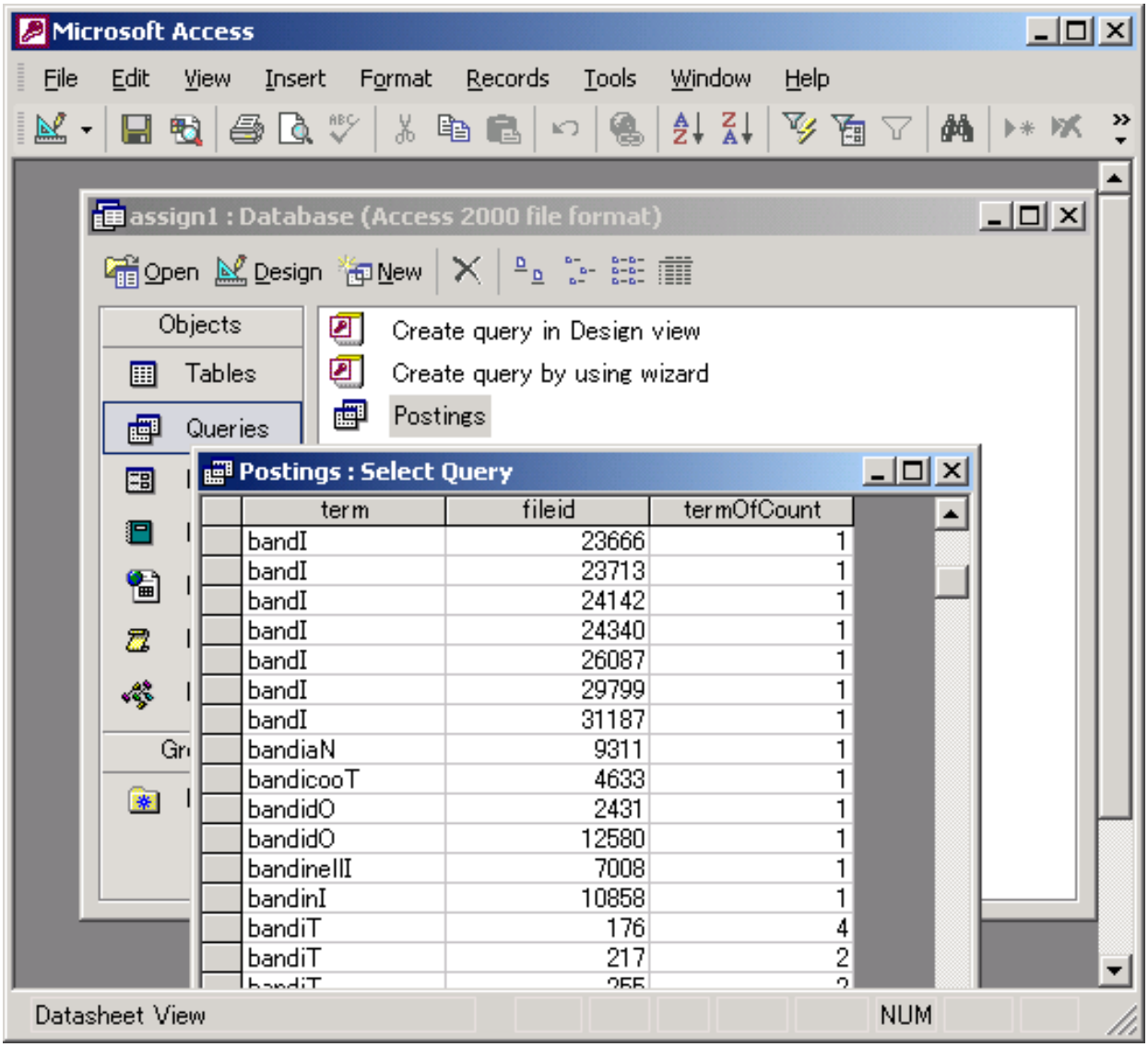
The file table has 64814 records.

Step 20: Create a query (called Postings) to count the within-document term frequency. First, you need to create a query as follows in SQL:

```
SELECT Post.term, Post.fileid, Count(Post.term) AS termOfCount
FROM Post
GROUP BY Post.term, Post.fileid;
```



Step 21: Compile the query:



Step 22: Based on the data (e.g. file.txt + post.txt or the MS Access file that stores these data) you obtained, you should develop an IR system using this information. There are several possible ways that you can develop the system as follows:

(a) use MS Access and possibly embedding some VBA;
(b) use MS Access as the database server and write a client to transfer the posting information to the client during retrieval. Your client program may be written in JAVA or C++ using some database connection utility like ODBC;
(c) develop a customized JAVA or C++ program that directly reads data from files.txt and post.txt (or some derived data). In this case, don't use any database package to help
(d) use another method but you need to make sure that the computing lab has the same package (e.g. using another database software) so that you can do the demonstration.

You are not allowed to use any open source search engine.

Step 23: When you input a query to the search engine, you need to stem the words in the query the same was as the index terms are stemmed. The stemmer program is already available in C++. This program is stored in stem.cpp and stem.h, based on the Porter stemming algorithm. To use this program in C++,

(a) declare that you are using this stemmer in the program by including the header file and declaring a stemmer object:

```
#include <stem.h>
stemmer StemProc;
```

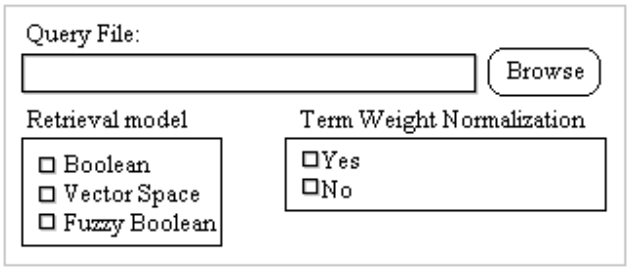
(b) call the member function `Stem`(.) to stem the word;

```
char word[1000];
strcpy(word, "organs")
StemProc.Stem(word); // word contains "orgaN"
```

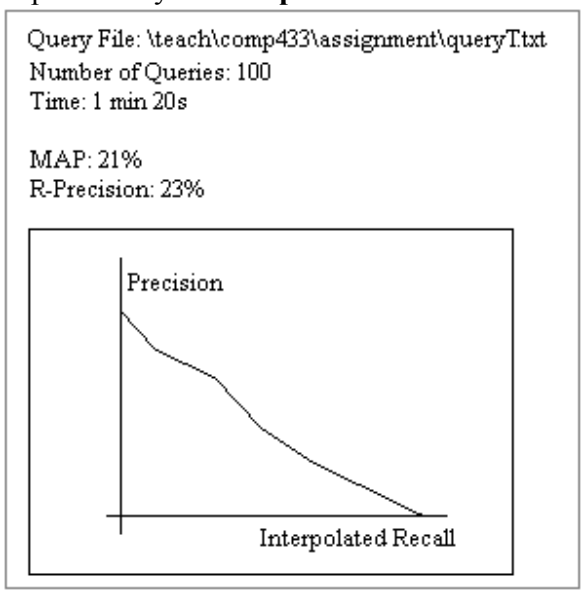
If you develop your JAVA program, you need to download a JAVA implementation of the Porter stemming algorithm from the web. In this case, you need to modify the JAVA program so that after stemming, the last character of the stemmed word is a capital letter (e.g., the last letter is in upper case for "orga**N**"). This is because our stemmer capitalized the last letter of the index terms. Otherwise, it will be difficult for you to match the index terms with the query terms.

Step 24: A possible interface for the system may be:

A possible system **input interface**:



A possible system **output interface**



NB: the TREC evaluation program (trec_eval.tar) produces the MAP, R-precision, interpolated recall-precision curve based in the ranked output of your search engine and the file (i.e., judgerrobust) containing the correct answers.