Design:

It is quite simple as kind of mentioned in the document provided. First the path is traversed, if it is a directory, in a recursive manner, or else file is processed directly.

TraverseFiles() is the function to get to files in a recursive way, ReadFile() is the function to process a particular file.

As and when there is a file encountered, the file is read character by character (in ReadFile() function), listing all the tokens found in the file.

The XML file is produced by OutputToXML() function.

Three linked lists are being used: IndexList, TokenList and DocTokenList.

IndexList is basically the inverted index for all the terms found in all the files. Every node in this will hold the token with a pointer to second list called TokenList. This TokenList is nothing but contains all the documents with the number of times that token is found in that document.

TokenList, as mentioned above, is the listing for the combination of document name and the number of times the token was found in that document. Every token entry in IndexList will have one of this list.

Third list called DocTokenList is kind of a temporary list being used to first scan a file completely, store all its terms and then update the global IndexList with all of these terms.

Flow is very simple, file is found, is tokenized, each token is pushed into DocTokenList, which is then traversed token by token and pushed into IndexList following all the sort criteria.

One Important point worth mentioning here is that, in the code, there is a limit considered on the name of the file, to be 50 chars max, it is a preprocessor defined, if required, it can be changed.

Time complexity:

It takes O(N) time to iterate from start to end of file to parse tokens. If there are M files in a directory it will take O(M*N) time.

Outputting the XML file takes O(N) time to write it.

Sorting tokens takes O(N) time as we are using a list and insertion takes place after sorting.

Same for files, sorting files takes O(N) time as we are doing sorting and insertion in a linear way.