```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <cstring>
#include <alloca.h>
#include <map>
#include <set>

using namespace std;

//removes all the nonalpha characters and replace them with spaces.
char* cleanUp(char* line, int lineLength){
    int i ;
    for(i=0; i<lineLength; i++){
        if(!isupper(line[i]) && !islower(line[i])){
            line[i] = ' ' ;

        }
    }
    return line ;
}



int main(int argc, char* argv[])
{

    map <string, set<int> > invertedIndex; //created a map with set Integers
    set <string> sortedKey ; //creates a string set which we are going to use to sort alphabetically
    char* pointer ;
    char* pointer1 ;
    char* str ;
    char* str1 ;
    char* newLine ;
    int lineSize ;
    string line ;
    string fileline ;
    int index = 0 ;
    set<int> currentWord ;

    ifstream inputText(argv[1]) ; //takes the second arg as filename


    while(inputText.is_open()){ //checks if the input.txt file is_open

        while(getline(inputText, line)){
            str = strdup(line.c_str()) ;
            pointer = strtok(str, " ") ; //space as delimeter
            while(pointer != NULL){ //while there are still tokens, next filedocument
                if(pointer != " "){ //if there is a file.
                    ifstream file(pointer) ; //opens up file
                    //process and put into map
                    while(getline(file, fileline)){ //gets the line inside textfile.
                        lineSize = fileline.size() ; //gets length of line
                        str1 = strdup(fileline.c_str()) ;
                        newLine = cleanUp(str1, lineSize) ; //cleaned up line

                        //tokenizes the cleanedup line.
                        pointer1 = strtok(newLine, " ") ;
                        while(pointer1 != NULL){ //while there are more words.

                            sortedKey.insert(pointer1) ; //inserts the string into sortedKey string set for later traversal output use

                            currentWord = invertedIndex[pointer1] ;
                            currentWord.insert(index) ;//inserts the document number into currentword.
                            invertedIndex[pointer1] = currentWord ; //inserts into map
```

```
67
68
69                              //test
70                              //cout << "Inserting word " << pointer1 << " from document
                                " << index << "!" <<endl ;
71                              pointer1 = strtok(NULL, " ") ; //points to next token
72                          }
73                      }//end while
74
75                      file.close() ;
76                      index++ ;//keeps track of filecount/index
77                  }
78                  pointer = strtok(NULL, " ") ; //reassigns pointer so it goes to the
                    next textfile.
79              }
80          }
81          inputText.close(); //closes file
82      }
83
84      set<string> :: iterator it; //creates an iterator of set<string> called it
85
86      for (it = sortedKey.begin(); it != sortedKey.end(); it++) {
87          string key = *it;
88          cout << key << ":" ; //prints out the key already sorted in set string
89
90          set<int> :: iterator iter; // creates an iterator for set<int>
91
92          for (iter = (invertedIndex.find(key)->second).begin(); iter != (invertedIndex.
            find(key)->second).end(); iter++) {
93              cout << " " << *iter ; //prints out the value using the set <string> above
                to search for.
94          }
95          cout << endl ;
96      }
97
98      return 0;
99  }
100
101
102
103
104
105
106
107
108
```