This assignment involves writing a C++ program to identify and count lines, words, and characters. For the purpose of this assignment, we define a “word” as a sequence of characters that are not whitespace characters. Words are delimited by one or more whitespace characters. A whitespace character, such as a space, tab, or newline, is any character for which the <cctype> function isspace() returns a value of true.

The operation of your program is controlled by the command line arguments that are passed to your program. Command line arguments consist of an optional set of flags arguments (a flag argument is an argument whose first character is a dash), followed by an optional list of filenames. If no filenames are specified, the program should read from standard input.

For each file specified (or, in the case of no file names specified, the standard input), the program must read the entire input and produce a count of characters, lines, and words. The output should be printed, one line per file, as follows:

Lines words characters filename

The numerical output (lines, words, characters) should be printed right justified in columns that are 12 characters wide. The filename should be printed left justified in the fourth column. In the case where the program in reading standard input, no filename should be printed.

If the program reads multiple files, then it should also provide a line at the end of the output with a total of all of the lines, words and characters in all of the files that were processed. The totals are tagged with the label “total”:

totalLines totalWords totalCharacters total

If the program is run with a flag in the form “-findchar=x”, then the program must also count the number of times it sees the character x in each file.

For example if I run the program by saying

program -findchar=m infile1 infile2

Then the output might be:

10 200 1000 infile1
20 150 1010 infile2
30 350 2010 total
m: 3 20 100 infile1
m: 0 0 0 infile2
m: 3 20 100 total

This means that the letter m appears on 3 different lines in infile1, in 20 different words, for a total of 100 times

If the program is run with a flag in the form “-findword=thisword”, then the program must count the number of times it sees the word thisword in each file.

For example if I run the program by saying

program -findword=this infile1

Then the output might be:
10 200 1000 infile1
this: 7 12 infile1

This means that the word this appears on 7 different lines for a total of 12 times.

The program should accept multiple -findchar= arguments and multiple -findword= arguments, in any order. The output should be counts, then all findchar cases in alphabetical order, then all findword cases in alphabetical order

The student will be provided with a test script, a set of test files, and a series of test cases.

There are several error cases that must be handled:

Error	Action
A file is not found	Print “File *filename* is not found” and stop
An argument is poorly formed (a known argument is missing the = sign, has nothing after the = sign, or (in the case of findchar, has more than one character after the =)	Print “Argument *arg* is poorly formed” and stop
An argument is not recognized	Print “Argument *arg* is not recognized” and stop
A flag is duplicated	Ignore the fact that a particular flag argument is duplicated. Only process each letter or word one time no matter how many times it is specified on the command line
A file name appears more than once	The file should be processed each time the name appears in the list

DUE DATES AND TEST CASES

On Feb 6, the following test cases will be due and must pass:

- Compile
- Count lines, words and characters in one file
- Correctly handle the file not found error case
- Detect poorly formed arguments
- Detect unrecognized arguments

On Feb 13, the following test cases will be due and must pass:

- All Feb 6 test cases
- Handle multiple files
- Handle reading from standard input
- Case of single findword

On Feb 20, the following test cases will be due and must pass:

- All Feb 13 cases
- Full implementation of findword
- Full implementation of findchar

The first two deliverables will have a 3 day grace period, where 10% of the grade for those deliverables is lost for every day of lateness.

The final deliverable will have a 7 day grace period, where 10% of the grade for the deliverable is lost for every day of lateness

Sample files and test cases will be provided and will automatically populate in Vocareum.