CS 280
Spring 2020
Programming Assignment 1

Part 1 due 2/26 (late tork accepted until noon Sunday 3/1)
Part 2 due 3/4 (late work accepted until noon Sunday 3/8)

For this assignment we will write a C++ program that serves as a very rudimentary word processor.

Our program will read input and perform some transformations, and generate output.

For this assignment, we use the following definitions:

- A “word” is a sequence of non-whitespace characters.
- An “empty line” is a line with no characters in it at all.
- A “blank line” is a line containing only one or more whitespace characters.
- The “line length” is the number of characters in each line that was output.
- A “paragraph” is a collection of lines. There are no blank lines or empty lines in a paragraph.
- An “indent” is spaces at the very beginning of the first line of a paragraph.

The program reads lines of input and prints out paragraphs.

The line length for each output line may be specified as a command line argument. If no command line argument for the line length is specified, the program should use a line length of 40.

The number of spaces to indent each paragraph may be specified as a command line argument. If no command line argument for indent is specified, the program should use an indent of 8.

Words in the output are separated by a single whitespace character. So, for example, the input “Hello world” should appear as “Hello world” in the output.

One or more consecutive empty or blank lines, or the end of file, should be interpreted as the end of a paragraph. Several consecutive empty or blank lines should cause a single empty line to be generated on output.

The output must be filled out to a length less than or equal to the line length. This may involve combining several lines of input into a single line of output.

For example, assume we have an input file like so:

This is
my favorite class.

Of all time!! Really, I swear!

If the indent is 5, and the line length is 80, then the output will be:

    This is my favorite class.

    Of all time! Really, I swear.

Notice that we combined the first two lines of input into a single line of output, we deleted the multiple spaces between This and is, we indented 5 spaces, and we combined multiple blank lines into a single line between paragraphs.

Whitespace at the beginning and end of a line of input should be ignored.

Note that the length of the output line should not exceed the line length. The only exception to this rule is if there is only a single word on the output line.

For example, suppose we have an indent value of 10, a line length of 15, and input as follows:

          Several
of us just looooooooooooooooooove   C++!

The output would be:

          Several
of us just
looooooooooooooooooove
C++!

Observe that we ignored the leading spaces in the first line of input. Note also that the first and third lines of output exceed the line length of 15; because there is only one word in the output line, this is acceptable. The second line of output does not have “looooooooooooooooooove” on the line because adding it would cause the output line to exceed the line length.

The program is instructed as to which options it should use by way of command line arguments.

Command line argument	Meaning
-LL=n	The output line length should be set to n. The value of n must be a positive integer greater than 0
-IN=n	The indentation should be set to n. The value of n must be a positive integer greater than 0
filename	Name of the file to be opened and read for input. A filename cannot begin with a dash.

If no filename is provided, the program should read from the standard input.

You are allowed to have multiple -LL and -IN flag arguments. If you do, the last one specified is used. So, for example, providing the command line arguments “-LL=5 -LL=20 -LL=100” results in a line length of 100.

The following error cases must be checked and handled:

Error case	How it is handled
More than one filename provided	Print “TOO MANY FILENAMES” and stop
Unable to open file	Print “UNABLE TO OPEN FILE filename” and stop
Unrecognized flag argument (a flag argument is an argument that begins with a dash)	Print “UNRECOGNIZED FLAG flag”, and stop
-LL or -IN flag does not have = after LL or IN	Print “MISSING = SIGN flag” and stop
-LL=n or -IN=n and n is not a positive integer greater than 0	Print “VALUE NOT INTEGER > 0 flag” and stop

When you go into Vocareum you will find a zip file of all the test case files, a README (which you should read) and a shell script called “runcase”

The test cases for part 1 test all of the error cases for opening files, all of the error cases for command line arguments, and a case with an empty file. Part 2 adds test cases for generating actual output.