CS280 Programming Assignment 1
Fall 2017
Part 1 due ~~Sep 25~~ Oct 2 before midnight, no later than ~~9/30~~ 10/7
Part 2 due ~~Oct 2~~ Oct 9 before midnight, no later than ~~10/7~~ 10/16

For this assignment we are going to gather statistics about words in files. Our goal is to write a program that reads the text from files and determines information about the longest word and longest line in each file.

For the purpose of this assignment, a “line” is defined to be all characters up to a newline; in other words, what is fetched by the getline() function. A “word” is defined to be a sequence of one or more non-whitespace characters. Words are separated by whitespace (space, tab, newline, etc).

The input to the assignment is provided as command line arguments. Command line arguments consist of optional “flag” arguments (a “flag” is an argument whose first character is a dash), followed by one or more file names. If file names are provided, the program should perform its operations on each file. If no file names are provided, the program should print the error message NO FILES and should terminate.

The default processing for each file is to read through the entire file, find the longest word and the longest line in the file, and generate output.

The program recognizes certain optional flag arguments that change the behavior of the program. Your program must recognize the arguments if they are provided and change the behavior of the program.

The flag arguments can appear in any order. There can be more than one of them specified. If duplicates of the same argument are specified, it’s ok to ignore the duplicate. The flags are:

- -m when printing the longest word in the file, print only the longest words that appear the most often
- -c in addition to printing the longest words and the length of the longest line, include, in parenthesis, a count of the number of times that the word/line appears
- -b consider multiple blanks between words on a line as a single blank for the purpose of calculating the length of a line

If a flag is provided that the program does not recognize, the program should print a line with the unrecognized flag followed by the string UNRECOGNIZED FLAG. The program should exit after printing this line.

If a filename provided on the command line cannot be opened, the program should print a line with the filename followed by the string FILE NOT FOUND. The program should continue with the next file after printing this line.

The format of the output for each file consists of a maximum of three lines

- A line with the filename followed by a colon
- A line with a sorted list of the words that are the longest words in the file, comma separated
- A line containing a number which is the length of the longest line in the file

If there are no words in the file, then the second line in the output (with the words) is skipped.

For example, suppose we have a text file named “test1” that contains this

```
fish
I like fish
Fish are yum
A big cat ate the rat and got fat
```

The output would look like this:

```
test1:
Fish, fish, like
33
```

The output when including the -m flag would look like this:

```
test1:
fish
33
```

The output when including the -c flag would look like this:

```
test1:
Fish(1), fish(2), like(1)
33(1)
```

The output when including the -c and the -m flag would look like this:

```
test1:
fish(2)
33(1)
```

Observe that the longest words are 4 characters long, so all of the 4 character words are printed in the output, in sorted order. Words are case sensitive: fish and Fish are different words.

For another example, suppose we have a text file named “test2” that contains this:

```
I like fish food, and you do too
Don’t look now your hair is blue
Don’t look now     your hair is blue
```

The output would look like this:

```
test2:
Don’t, food,
36
```

The output when including the -m flag would look like this:

```
test2:
Don’t
36
```

The output when including the -c flag would look like this:

```
test2:
Don’t(2), food,(1)
36(1)
```

The output when including the -c and the -m flag would look like this:

```
test2:
Don’t(2)
36(1)
```

The output when including the -c and the -b flag would look like this:

```
test2:
Don’t(2), food,(1)
33(3)
```

The test cases for the assignment are as follows:

Part1 (8 points):

- Compiles
- Nofiles (i.e. runs the program with no file names)
- Badfiles (i.e. runs the program with no valid file names specified)
- Invalidflag (i.e. runs the program with an invalid flag name)
- Onefile (i.e. runs the program with a single input file)
- Manyfile (i.e. runs the program with many files)
- Onecount (Onefile case with -c arg added)
- Manycount (Manyfile case with the -c arg added)

Part2 (16 points): all of Part1, plus:

- Onemax (Onefile case with the -m flag)
- Manymax (Manyfile with the -m flag)
- OneMC (onefile with -m and -c)
- ManyMC (manyfile with -m and -c)
- Oneblank (onefile with -b)
- Manyblank (manyfile with -b)
- Oneall (onefile, all flags)
- Manyall (manyfile, all flags)

Total assignment is worth 24 points.