Project 2

Due Friday by 11:59pm **Points** 100 **Available** Feb 20 at 12am - Mar 9 at 12:15am 17 days

Project 2

In this project we are going to be expanding into the rest of the ways to structure our programs. We will be focusing on loops and functions in addition to selection statements. So you'll be required to use loops and functions in this project.

Details

For this project we'll be reading in a tab delimited file. The file will be information about countries around the globe. Then we'll be processing a series of commands about the file. On each command you'll be reading the file that was given by the command file. You won't be storing the file information in any sort of structure between commands, i.e. no using arrays, vectors, lists, etc.

The project description is long, you might have to read it again to understand clearly. Once you've understood, to get started on coding, watch the three videos Using Files with Functions, Reading Tab Delimited Data Part 1 and Reading Tab Delimited Data Part 2. You can access this in **Helpful Videos.**

Commands

There will be 3 commands we will be dealing with. One of the commands will have 3 versions, so we'll have a total of 5 commands if you want to count them that way.

The command file will start with a line that has the word `Data File:` and then a file name. That file name is the name of the data file you'll be using for the commands to do its searching. You are **not** to read and store that information in any sort of array, list, vector, etc.

```
size - the size command will have one of 3 subcommands after it.
    average - if the word average comes after the size command, then you'll calculate the average size of the countrie
s that are included in the input file.
    smallest - if the word smallest comes after the size command, then you'll find and display the smallest of the cou
ntries that are included in the input file.
    largest - if the word largest comes after the size command, then you'll find and display the largest of the countr
ies that are included in the input file.

country - the country command will be followed by a string that could represent a country code from the file.  You'll
search through the input file and if you find the given country code, then you'll display the information about that c
ountry.

neighbors - the neighbors command will be followed by a string that could represent a country code from the file.  Yo
u'll search through the input file and if you find the given country code in the neighbors list for a country that's i
ncluded in the input, then you'll display the information about that country.
```

Each of the commands has the possibility for nothing to be found and displayed. So there will be responses for those. Look lower to see what to do in those case.

Here's a sample of what the input could for the commands could look like

```
Data File: countryInfo.txt
size average
size largest
size smallest
country US
neighbors US
country NONE
neighbors WHO
```

So this shows all of the possible commands including some that should probably produce negative results for not finding a country or a neighbor.

It is possible that the command file and/or the data file are empty.

Details

Details and outputs for each of the commands are shown below:

```
Data File: <filename> - this will be the first line in the command file.  It will start with the words "Data File:" an
d be followed with a file name that you are to use to read the data when you are processing the rest of the commands.
```

You should echo the words "Data file:" and the file that was given. This should always be the first line in the output file.

```
size <subcommand> - the size command has 3 possible subcommands, average, largest, smallest.
    average - this will read every country that is included in the data file and produce the average size.  The outpu
t should be the following:

Command: size average
    Sum: 150012539.69
    Country count: 252
    Average Size: 595287.86
```

The first line is an echo of the command and subcommand. The next line starts with "Sum:" and is followed by the sum of the areas. The 3rd line starts with the word "Country count:" and is followed by how many countries are included in the sum. The 4th line starts with "Average Size:" followed by the average of the sum/count.

```
 largest - this will search the given data file for the largest country in the file.  If one is found, then all of the
details about the country will be displayed as in this example:

Command: size largest
        ISO:                    RU
        ISO3:                   RUS
        ISO Numeric:            643
        FIPS:                   RS
        Country:                Russia
        Capital:                Moscow
        Area:                   17100000.00
        Population:             140702000
        Continent:              EU
        TopLevelDomain:                 .ru
        Currency Code:          RUB
        Currency Name:          Ruble
        Phone:                  7
```

```
        Postal Code:            ######
        Postal Code Regex:      ^(\d{6})$
        Languages:              ru,tt,xal,cau,ady,kv,ce,tyv,cv,udm,tut,mns,bua,myv,mdf,chm,ba,inh,tut,kbd,krc,ava,sah,
nog
        Geonameid:              2017370
        Neighbors:              GE,CN,BY,UA,KZ,LV,PL,EE,LT,FI,MN,NO,AZ,KP
        Equivalent FIPS Code:
```

In this example, the country didn't have some of the fields filled in, and they are display as blank.

```
    smallest - this will search the given data file for the smallest country in the file.  If one is found, then all t
he details about the country will be displayed as in this example:

Command: size smallest
        ISO:                    UM
        ISO3:                   UMI
        ISO Numeric:            581
        FIPS:
        Country:                United States Minor Outlying Islands
        Capital:
        Area:                   0.00
        Population:             0
        Continent:              OC
        TopLevelDomain:                 .um
        Currency Code:          USD
        Currency Name:          Dollar
        Phone:                  1
        Postal Code:
        Postal Code Regex:
        Languages:              en-UM
        Geonameid:              5854968
        Neighbors:
        Equivalent FIPS Code:
```

In this example, the country didn't have some of the fields filled in, and they are displayed as blank

```
country <ISO> - this command will be followed by a country iso code.  The file is to be searched and if the given iso
code is found, then the country details are displayed as in this example:

Command: country US
        ISO:                    US
        ISO3:                   USA
        ISO Numeric:            840
        FIPS:                   US
        Country:                United States
        Capital:                Washington
        Area:                   9629091.00
        Population:             310232863
        Continent:              NA
        TopLevelDomain:                 .us
        Currency Code:          USD
        Currency Name:          Dollar
        Phone:                  1
        Postal Code:            #####-####
```

```
        Postal Code Regex:      ^\d{5}(-\d{4})?$
        Languages:              en-US,es-US,haw,fr
        Geonameid:              6252001
        Neighbors:              CA,MX,CU
        Equivalent FIPS Code:

If the iso code is not found, then a message is displayed.  For example, if the command was country NONE

Command: country NONE
    Country NONE not found.

So the ISO code can be longer than 2 characters and we still do the search.
```

```
neighbors <ISO> - this command will search the given data file and look in the neighbor list for the given ISO code.
For each country where the ISO code was found, the details for the neighbor country is found along with a count of th
e neighbor. The first found neighbor is 1, the second is 2, and so on.

Command: neighbors US
Neighbor 1
        ISO:                    CA
        ISO3:                   CAN
        ISO Numeric:            124
        FIPS:                   CA
        Country:                Canada
        Capital:                Ottawa
        Area:                   9984670.00
        Population:             33679000
        Continent:              NA
        TopLevelDomain:                 .ca
        Currency Code:          CAD
        Currency Name:          Dollar
        Phone:                  1
        Postal Code:            @#@ #@#
        Postal Code Regex:      ^([ABCEGHJKLMNPRSTVXY]\d[ABCEGHJKLMNPRSTVWXYZ]) ?(\d[ABCEGHJKLMNPRSTVWXYZ]\d)$
        Languages:              en-CA,fr-CA,iu
        Geonameid:              6251999
        Neighbors:              US
        Equivalent FIPS Code:


    This is the first of the found neighbors for the US.  There should be two more for the complete results.

    If no neighbors are found then a message is display, for example, if the command was neighbors WHO

Command: neighbors WHO
    Country WHO has no neighbors.
```

For each command the first thing the command should do is echo the command and any arguments.

You can see that many of the commands produces an output of all the details about a country. This is an excellent idea of a function.

Data File

The data file may potentially contain any number of lines at the top that are comments. Comments are indicated by any line that starts with a #. Any line that starts with a # should be ignored and are not considered part of the input data, even if that line would normally be a country data record.

The records are made up of several fields, any of which could be blank. They are

1. ISO
2. ISO3
3. ISO-Numeric
4. fips
5. Country
6. Capital
7. Area(in sq km) - this should be a double not an int
8. Population
9. Continent
10. topleveldomain
11. CurrencyCode
12. CurrencyName
13. Phone
14. Postal Code Format
15. Postal Code Regex
16. Languages
17. geonameid
18. neighbors
19. EquivalentFipsCode

Yes, that's a lot of fields, and yes I want you to read them all.

Description of some of the fields:

1. ISO - that's the country's ISO abbreviation. It's two letters long. It's what we are using when we are searching for a country. For example, the United States would be US not USA.
2. Area(in sq km) - that's the area of the country in square kilometers. We'll be using this to calculate average, smallest and largest countries.
3. neighbors - this is a comma separated list of ISO country codes. For example, the US's neighbor list is: CA,MX,CU. I'll leave it to you to figure out what countries those are. (Hint: this program would find those countries if you do it correctly.)

The rest of the data is just coming along for the ride. When we display a country, we will display the entire record, e.g. every field, all 19, even if it is blank.

I'll be posting the complete countryInfo.txt and some samples for you to use. Some of the samples will have some of the countries commented out. Your program should treat those countries as not being in the file and they are not searched.

Ouput

If we use the countryInfo.txt file and the commands shown above this would be the output:

```
Data file: countryInfo.txt
Command: size average
        Sum: 150012539.69
        Country count: 252
        Average Size: 595287.86
Command: size largest
        ISO:                    RU
        ISO3:                   RUS
        ISO Numeric:            643
        FIPS:                   RS
        Country:                Russia
        Capital:                Moscow
        Area:                   17100000.00
        Population:             140702000
        Continent:              EU
        TopLevelDomain:                 .ru
        Currency Code:          RUB
        Currency Name:          Ruble
        Phone:                  7
        Postal Code:            ######
        Postal Code Regex:      ^(\d{6})$
        Languages:              ru,tt,xal,cau,ady,kv,ce,tyv,cv,udm,tut,mns,bua,myv,mdf,chm,ba,inh,tut,kbd,krc,ava,sah,
nog
        Geonameid:              2017370
        Neighbors:              GE,CN,BY,UA,KZ,LV,PL,EE,LT,FI,MN,NO,AZ,KP
        Equivalent FIPS Code:
Command: size smallest
        ISO:                    UM
        ISO3:                   UMI
        ISO Numeric:            581
        FIPS:
        Country:                United States Minor Outlying Islands
        Capital:
        Area:                   0.00
        Population:             0
        Continent:              OC
        TopLevelDomain:                 .um
        Currency Code:          USD
        Currency Name:          Dollar
        Phone:                  1
        Postal Code:
        Postal Code Regex:
        Languages:              en-UM
        Geonameid:              5854968
        Neighbors:
        Equivalent FIPS Code:
Command: country US
        ISO:                    US
        ISO3:                   USA
        ISO Numeric:            840
        FIPS:                   US
        Country:                United States
        Capital:                Washington
        Area:                   9629091.00
```

```
        Population:             310232863
        Continent:              NA
        TopLevelDomain:                 .us
        Currency Code:          USD
        Currency Name:          Dollar
        Phone:                  1
        Postal Code:            #####-####
        Postal Code Regex:      ^\d{5}(-\d{4})?$
        Languages:              en-US,es-US,haw,fr
        Geonameid:              6252001
        Neighbors:              CA,MX,CU
        Equivalent FIPS Code:
Command: neighbors US
Neighbor 1
        ISO:                    CA
        ISO3:                   CAN
        ISO Numeric:            124
        FIPS:                   CA
        Country:                Canada
        Capital:                Ottawa
        Area:                   9984670.00
        Population:             33679000
        Continent:              NA
        TopLevelDomain:                 .ca
        Currency Code:          CAD
        Currency Name:          Dollar
        Phone:                  1
        Postal Code:            @#@ #@#
        Postal Code Regex:      ^([ABCEGHJKLMNPRSTVXY]\d[ABCEGHJKLMNPRSTVWXYZ]) ?(\d[ABCEGHJKLMNPRSTVWXYZ]\d)$
        Languages:              en-CA,fr-CA,iu
        Geonameid:              6251999
        Neighbors:              US
        Equivalent FIPS Code:
Neighbor 2
        ISO:                    CU
        ISO3:                   CUB
        ISO Numeric:            192
        FIPS:                   CU
        Country:                Cuba
        Capital:                Havana
        Area:                   110860.00
        Population:             11423000
        Continent:              NA
        TopLevelDomain:                 .cu
        Currency Code:          CUP
        Currency Name:          Peso
        Phone:                  53
        Postal Code:            CP #####
        Postal Code Regex:      ^(?:CP)*(\d{5})$
        Languages:              es-CU
        Geonameid:              3562981
        Neighbors:              US
        Equivalent FIPS Code:
Neighbor 3
        ISO:                    MX
        ISO3:                   MEX
        ISO Numeric:            484
        FIPS:                   MX
        Country:                Mexico
        Capital:                Mexico City
```

```
        Area:                   1972550.00
        Population:             112468855
        Continent:              NA
        TopLevelDomain:                 .mx
        Currency Code:          MXN
        Currency Name:          Peso
        Phone:                  52
        Postal Code:            #####
        Postal Code Regex:      ^(\d{5})$
        Languages:              es-MX
        Geonameid:              3996063
        Neighbors:              GT,US,BZ
        Equivalent FIPS Code:
Command: country NONE
        Country NONE not found.
Command: neighbors WHO
        Country WHO has no neighbors.
```

Samples zip folder contains sample commands, data and output files. Check your code against those sample files as well to make ensure correctness. Download the Revised samples folder. **RevisedSamples_Project2.zip**

Requirements

Here are the requirements the TAs will be using to grade the remaining 25 points for project 2

1. You must name your header file `countryInfo.h` and it must contain a declaration for a function with the following signature: `void countryInfo( string input, string output );`
2. You may **not** use an array, list, vector, or other STL container class to store the contents of the file. The file must be read for each of the commands. Yes I know this is inefficient. That's not the point of this project. Get it right.
3. You may not use global variables of any kind.
4. You **must** write a function for each of the command. Size may handle the 3 sub functions in the 1 function or you may write 3 separate functions for the 3 separate size commands. That means you write the countryInfo function, a size function, a country function and a neighbors function.
5. I would recommend you write a function that reads 1 line of the given input data file and returns all the pieces. It will help. It is not required.
6. You follow the Codestyle Guidelines listed in Canvas.
7. You must use a loop in your code. (Don't worry you will probably have more than one.)
8. You may not use structs or (user created) classes; strings, iostream, ones we've been using are fine.

Naming

When you name your cpp files, do not use spaces or punctation marks. The . for the .cpp is fine. For example you could call this project2.cpp or countryInfo.cpp. The name of the cpp is up to you. The header file must be named as it states in the requirements.

Grading

Zip up your countryInfo.h and countryInfo.cpp files and submit them to Web-CAT for grading.

You have 12 submissions.

Reminder: **Projects are individual work.** Your project code submission will be run through a system know as Moss that was developed at Berkeley. It's purpose is to compare code for similarity and attempts to detect cheating. Cases of Cheating will be reported to the Honor Court

Due

This is due March 8. That is the Friday before Spring Break starts and the Friday after the last day to drop. So if you feel this project is beyond you, then you should take that into consideration. The next 2 projects will be harder.