**Project Goals:**

1. Learn to create class structure in C++
2. Create an array of objects in C++
3. Search and perform operations on the array of objects

**Project Description:**

The input csv file for this project consists of rows of data that deals with COVID-19 cases and deaths per day for each county in every state in the United States. Here is an example,

date,county,state,fips,cases,deaths

2020-01-21,Snohomish,Washington,53061,1,0

2020-01-22,Snohomish,Washington,53061,1,0

2020-01-23,Snohomish,Washington,53061,1,0

For the purposes of this project, we will assume that the following are char\* data types: date, county, and state. FIPS (unique identifier for each county) along with cases and deaths are int data types. Please note the comma delimiter in each row. You need ­­­to carefully read each field knowing that you will have a comma.

You will use redirected input (more later) to read an input txt file that contains the following:

counts //number of data entries in the csv file

Filename.csv //this is the file that contains the covid-19 data

Command //details of what constitutes a command is given below

Command

Command

….

Your C++ program will read the counts value on the first line of the txt file, which represents the number of data entries in the csv file. (Note - the first line of the csv file contains descriptive variable fields, so there will be a total of [number of data entries + 1] lines in the csv file). Then, on the second line, it will read the Filename.csv and open the file for reading (more on how to do this in C++). After you open the file, you will read the data from each row of the csv file and create a COVID19 object. The COVID19 class is given below. You need to implement all of the necessary methods.

class COVID19 {

protected:

char\* date;

char\* county;

char\* state;

int fips;

int cases;

int deaths;

public:

COVID19 (); //default constructor

COVID19 (char\* da, char\* co, char\* s, int f,

int ca, int de); //initializer

display ();

//write all accessors and other methods as necessary

};

After your write the above class you will write the following class:

class COVID19DataSet {

protected:

COVID19\* allData;

int count; //number of COVID19 objects in allData

int size; //maximum size of array

public:

COVID19DataSet (); //default constructor

COVID19DataSet (int initSize);

void display ();

void addRow (COVID19& oneData);

int findTotalCasesByCounty (char\* county, char\* state);

int findTotalDeathsByCounty (char\* county, char\* state);

int findTotalCasesByState (char\* state);

int findTotalDeathsByState (char\* state);

int findTotalCasesBySateWithDateRange (char\* state,

char\* startDate, char\* endDate);

int findTotalDeathsBySateWithDateRange (char\* state,

char\* startDate, char\* endDate);

~COVID19(); //destructor

//other methods as deem important

};

The structure of the main program will be something like this:

#include <iostream>

using namespace std;

// Write all the classes here

int main () {

int counts; // number of records in Filename.CSV

int command;

COVID19 oneRow;

//read the filename, for example, Filename.csv

//open the Filename.csv using fopen (google it for C++ to find out)

//assume that you named this file as myFile

//read the first integer in the file that contains the number of rows

//call this number counts

COVID19DataSet\* myData = new COVID19DataSet (counts);

for (int i=0; i < counts; i++) {

//read the values in each row

//use setters to set the fields in oneRow

(\*myData).addRow (oneRow);

} //end for loop

while (!cin.eof()) {

cin >> command;

switch (command) {

case 1: {

//read the rest of the row

(\*myData).findTotalCasesByCounty (county, state);

break;

}

case 2: {

//do what is needed for command 2

break;

}

case 3: {

//do what is needed for command 3

break;

}

case 4: {

//do what is needed for command 4

break;

}

case 5: {

//do what is needed for command 5

break;

}

case 6: {

//do what is needed for command 6

break;

}

default: cout << “Wrong command\n”;

} //end switch

} //end while

delete myData;

return 0;

}

**Input Structure:**

The input txt file will have the following structure – I have annotated here for understanding and these annotations will not be in the actual input file. After the first two lines, the remaining lines in the input txt file contains commands (one command per line), where there can be up to 6 different commands in any order with any number of entries. The command is indicated by an integer [1 to 6] and can be found at the beginning of each line.

counts // Number of data entries in the csv file

Covid-19-Data-csv // Name of the input file that contains the actual Covid-19 data by county and state

1 Cleveland, Oklahoma //command 1 here is for findTotalCasesByCounty

1 Walla Walla, Washington

1 San Francisco, California

1 Tulsa, Oklahoma

2 Oklahoma, Oklahoma //command 2 here is for findTotalDeathsByCounty

2 Miami, Ohio

2 Miami, Oklahoma

3 Oklahoma //command 3 here is for findTotalCasesByState

3 North Carolina

4 New York //command 4 here is for findTotalDeathsByState

4 Arkansas

5 Oklahoma 2020-03-19 2020-06-06 //5 here is for findTotalCasesBySateWithDateRange

6 New York 2020-04-01 2020-06-06 //6 here is for findTotalDeathsBySateWithDateRange

**Redirected Input**

Redirected input provides you a way to send a file to the standard input of a program without typing it using the keyboard. To use redirected input in Visual Studio environment, follow these steps: After you have opened or created a new project, on the menu go to project, project properties, expand configuration properties until you see Debugging, on the right you will see a set of options, and in the command arguments type “< **input filename**”. The < sign is for redirected input and the **input filename** is the name of the input file (including the path if not in the working directory). A simple program that reads character by character until it reaches end-of-file can be found below.

#include <iostream>

using namespace std;

//The character for end-of-line is ‘\n’ and you can compare c below with this

//character to check if end-of-line is reached.

int main () {

char c;

cin.get(c);

while (!cin.eof()) {

cout << c;

cin.get(c);

}

return 0;

}

*C String*

A string in the C Programming Language is an array of characters ends with ‘\0’ (NULL) character. The NULL character denotes the end of the C string. For example, you can declare a C string like this:

char aCString[9];

Then you will be able to store up to 8 characters in this string. You can use **cout** to print out the string and the characters stored in a C string will be displayed one by one until ‘\0’ is reached. Here are some examples:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | cout result | Length |
| u | s | e | r | n | a | m | e | \0 | username | 8 |
| n | a | m | e | \0 |  |  |  |  | name | 4 |
| n | a | m | e | \0 | 1 | 2 | 3 | 4 | name | 4 |
| \0 |  |  |  |  |  |  |  |  | (nothing) | 0 |

Similarly, you can use a for loop to determine the length of a string (NULL is NOT included). We show this in the following and also show how you can dynamically create a string using a pointer

char aCString[] = "This is a C String."; // you don’t need to provide

// the size of the array

// if the content is provided

char\* anotherCString; // a pointer to an array of

// characters

unsigned int length = 0;

while( aCString[length] != '\0')

{

length++;

}

// the length of the string is now known

anotherCString = new char[length+1]; // need space for NULL character

// copy the string

for( int i=0; i< length+1; i++)

anotherCString[i] = aCString[i];

cout << aCString << endl; // print out the two strings

cout << anotherCSring << endl;

delete [] anotherCString; // release the memory after use

You can check <http://www.cs.bu.edu/teaching/cpp/string/array-vs-ptr/>, other online sources or textbooks to learn more about this.

**Output Structure**

Stay tuned for the exact format in which the output of your program should be formatted. For now, it is recommended to start working on reading in the input files, storing the data, and accessing the data based on the given commands.

**Constraints**

1. In this project, the only header you will use is #include <iostream> and using namespace std.