Term Project –Fall 2020
COVID-19 tracking System

Grade weight: 15%

Phase 1Due Date: Tuesday 27th /October/2020
Phase 2 Due Date: Saturday 21st /Nov/2020

Problem Description

A task is assigned to you; using the knowledge you have in C++ from the course CMPS151. The task is to create a small-computerized system that takes care of COVID-19 confirmed cases and deaths.

The system has four main operations; adding new cases, deleting, editing a case, and producing different reports or lists. The system you want to develop is an application that can track data that is similar to the data shown in the following table:

Country Code	Country	Total Cases	New cases (24 hours)	Total Deaths	New Deaths
1	USA	7244184	53629	208440	895
2	Mexico	715457	5408	75439	490
3	Ireland	33995	319	1797	3
4	India	5901571	85468	93410	1093
5	Turkey	311455	7858	7858	73

The main tasks that the system will be performing are as follows:

1. Add a record of a new country
2. List the information of all countries
3. Search for a specific country record based on the country code
4. Display a report of all countries with total number of cases higher than a specified value from user.
5. Update a record based on a specific field (new cases, new deaths)
6. Delete a record of a country
7. Functions related to display (green, yellow, and red) countries based on specific conditions
8. Exit

Requirements:

Phase1: Within the main function do the following: [In this phase, you are expected to use the following concepts: input/output, selection statements, loops, and files in C++].

You are required to develop the Covid'19 tracking System (**first 4 functions**). In this phase you will deal with *.txt file. The file is called "covid19_information.txt". In covid19_information.txt, you are expected to save and read from all the countries' records as shown in the table above.

The system should perform the following operations:

1. **Add a record of a new country:** that asks the user to input data of a new country and then add the country record to "covid19_information.txt" file.
   Note:
   - The country code must be a unique value for each country.
   - Use file reading refernce: http://www.cplusplus.com/doc/tutorial/files/
2. **List the information of all countries:** in a tabular format all the information about each country record. [hint: use iomanip library].
3. **Search for a specific country:** Ask the user to enter the country's code, and then display the country record on the screen.
   Note:
   - If the country's code does not exist in the file, then you need to inform the user about that, and return back for the main menu.
4. **Display a report of all countries with total number of cases higher than a specified value from user:** Ask the user to enter a total cases value, and then display all countries' records that has total cases higher or equal than the entered value.
   Note:
   - If the entered value is the highest value compared to the total cases in all countries, then display a message to inform the user about that.

An example of how the system works is shown below:

```
 COVID'19 tracking System

             [1] Add a record of a new country
             [2] List the information of all countries
             [3] Search for a specific country
             [4] Display a report of all countries with total
             number of cases higher than a specified value from
             user

             [8] Exit
    Enter your choice: 1
 Add a record of a new country:
========================
Enter country name: Qatar
Enter total cases: 124650
Enter number of New cases : 220
Enter number of deaths: 212
Enter number of new deaths: 2

.....You made a valid country record, and record is saved.

COVID'19 tracking System

             [1] Add a record of a new country
             [2] List the information of all countries
             [3] Search for a specific country
             [4] Display a report of all countries with total
             number of cases higher than a specified value from
             user

             [8] Exit



    Enter your choice: 8
 Bye, see you again later!
```

Phase 2: [You will be required to re-organize your code into functions and use arrays and write new functions].

In this phase, you will be continuing in the same project that you created in phase 1 and add three more functionalities. You will also be required to re-organize your code into functions.

- Move the code that you created in phase 1 into different functions. For instance, the code that add new country record should be moved to a function called "**addNewCountry()",** The code that displays the **menu** items should be moved to a function called "**menu**" and so on… Move

all the first 4 menu items into their functions. If one function can handle two different menu items, then create a single function and reuse it.

Implement the following new functionalities

1. **Update a record based on a specific field (new cases, new deaths):** This function can change the record for a certain country. Hence, write a function called **updateCountryRecord()** that takes as parameter the **country code.**

   Notes:

   - If the passed country code does not exist in the file, then inform the user about that, and return back to main menu.
   - Make a sub menu to ask the user what he/she would like to update (new cases, new deaths). If user chose new cases, then you will allow him/her to enter the number of new cases to update the old new cases filed in the record. In addition don't forget to update the total cases depending on that.

   Example : If user enters new cases to be 250

	Country Code	Country	Total Cases	New Cases (24 hours)	Deaths	New Deaths
Before	2	Qatar	**124650**	**220**	212	2
After	2	Qatar	**124900**	**250**	212	2

   The same idea for new deaths can be used.

   - Remember to save the changes in the file.

2. **Delete a record of a country:** Write a function called **deleteCountry**() that takes the **country code**. Then you should delete the country record from the "covid19_information.txt". file.

   Note:

   - If the country code does not exist in the then inform the user about that, and return back to main menu.

3. **Functions related to display (green, yellow, and red) countries based on specific conditions:** Using functions and arrays classify the countries into three categories based on the following conditions:

Condition	Category
$(\boldsymbol{new\ cases} \geq \mathbf{50000})\ \boldsymbol{and}\ (\boldsymbol{new\ deaths} \geq \mathbf{800})$	**Red**
$(\mathbf{50000} > \boldsymbol{new\ cases} \geq \mathbf{5000})\boldsymbol{and}\ (\mathbf{800} > \boldsymbol{new\ deaths} \geq \mathbf{70})$	**Yellow**
$(\boldsymbol{new\ cases} < \mathbf{5000})\ \boldsymbol{and}\ (\boldsymbol{new\ deaths} < \mathbf{800})$	**Green**

   Then display all countries codes, names with their classifications on the screen. In addition write those information (countries' codes, countries' names, classification color) into a file called " calssifications.txt".

Implementation Hint:

1. Use **arrays** for the three above functionalities. First, read all the values from files into arrays then do the operation on those arrays. After that, write the content of the arrays back to the files.
2. You can use a function for reading, and another function for writing. Then you ca call them based on your need.

Important:

1. Properly format your output.
2. Validate the user's input.
3. Whenever possible, reuse functions.
4. The project's main function should provide a menu that will allow the user to choose one of the tasks. The program exits when the user selects option 7.

Evaluation Criteria:

The grade for the project is **15 points** distributed as follows:

1. **Phase 1: (30%)**
   Phase 2: (30%)
   In each phase the following will be graded:
   1. Correct implementation of all the requirements using appropriate C++ programming techniques. Testing should be done to ensure that the results produced by the program match the expected results.
   2. Program Structure and Efficiency depending on the phase work:
      - Use of functions and re-use of functions whenever possible. (phase 2)
      - Use of arrays. (phase 2)
      - Use of loops. (both phase 1, and 2)
      - Use of files. (both phase 1, and 2)
      - Handling user input validations: e.g. menu choices, values entered …etc.
      - Code quality:
        a. Using meaningful identifiers.
        b. In-program comments, where necessary.
        c. White space and proper indentation.
        d. Informative input prompts.
2. **Discussion and demo of the project (40%):**
   Students should demo their program, explain how it works and answer questions about the implementation. Each team member must contribute to the discussion and demo. **If the student does not come to the discussion, then he/she will take ZERO in the project**. **Each student will be asked about any part in the project.**
3. **Copying and/or plagiarism (-100%)**
4. **The program does not compile (-50%)**

5 **In case of late submission, (-10%) for each day of delay. Only two days are allowed.**

Guidelines and Submission Instructions:

- **Each team will have 3 members**
- Team members are required to meet regularly for discussion and workload distribution (Through MS- teams channels in your group).
- Turn in your programs (as .cpp files **and as a copy in** .doc file).
- Each team should submit **only** one electronic submission
- **It is the right of the instructor to test the students undersetting of the project in away they see fit during the demo and discussion session. So, a project that is 100% working might be may be graded (-100%) due to student not being able to explain a functionality they have implemented**