TASK ONE

The algorithm maintains a series of initial variables and calculates the answers to all questions within a single *for-loop* when reading from the *covid\_data.csv* file. The population data is read into a dictionary before the loop in order to later calculate the *rates*

1. Which country has the highest number of infections to date? How many infections have been recorded in that country?
2. Which country has the second highest number of infections to date? How many infections have been recorded in that country?
3. Declare four variables to hold the *highest number of infections* and its *corresponding country*, the *second highest number of infections* and its *corresponding country*.
4. Read first country values from file and assign to the initial variables above variables.
5. Loop through the remaining countries in the file. When the sum of a country’s *new confirmed cases* is larger than the *highest number of infections*, update *highest number of infections* with the *new confirmed cases* . The previous *highest number of infections* becomes the *second highest number of infections*. The countries are also updated accordingly.

[CAN INLCUDE THE PSEUDOCODE HERE, OPTIONAL]

1. Which country has the highest infection rate (ratio of number of infections to population) to date? What is the infection rate?
2. Which country has the highest death rate (ratio of number of deaths to number of infections)? What is the death rate for that country?
3. Assign the value 0 and the first country read from file to the initial variables that will hold the *highest infection (and death) rate* and their countries accordingly.
4. Loop through the countries, calculate the infection (and death) rates for *each* country. When a l*arger infection (and death) rates* are found, update the *highest infection (and death) rates accordingly as well as their corresponding* countries.

[CAN INCLUDE PSEDOCODE HERE, OPTIONAL]

1. What is the overall death rate (ratio of number of deaths to number of infections) for COVID-19?
2. Initial variables, *total\_deaths* and *total\_infections* are initialized to value 0.
3. During the loop, the variables are updated by adding the infections and death rates of *each* country as the are calculated (c) and (d) above.
4. The overall death rate is calculated at the end of the loop as *total\_deaths* divided by *total\_infections.*

The correlation coefficient is calculated among the recent 7 (1 week data) data points for each each, and the countries with the positive correlation coefficient are retrieved as the countries with positive trend while those with negative correlation are retrieved as countries with negative trend. Among all the countries with the positive trends, the country with the highest positive correlation has the steepest increase while the country with the lowest negative correlation has the steepest decrease.

We generated integers from 1 - 7 as the x-values while the recent 1 week data as the y-values in order to calculated the correlation.

To find the country whose number of infections per day peak the earliest, we run a loop on the 1 week recent data on the infections per day for each country, and compare each item to the adjacent item in the list, and return item if its greater than its adjacent item. This becomes the peak item, and its data is retrieved. Each of these dates are compared and the earliest date together with its corresponding country returned.

TASK TWO

Our approach for solving this task uses pattern matching. We use the KMP string matching algorithm to find the pattern (partial time series data) from the movie data CSV file. We first read the new infection cases column to a list (e.g confirmed cases) with their corresponding dates and countries  in another list (e.g info. Info contains tuples of country and date). Also, we read the partial time series data into a list (e.g partial list). Given the confirmed cases list and the partial list, we use KMP algorithm to search for partial list in confirmed cases list and returns the index at which the partial list occurred in confirmed list. We then use the index found to find the country and the starting date from the info list.

 The main idea of the KMP algorithm is that whenever we have a mismatch, we don't discard what has been matched already. Instead, we leverage the fact that we already know some of the characters in the text of the next window. The algorithm skips matching characters in the next widow of the pattern with the next window of the confirmed list that we know will match and match the other characters to decide whether the current window matches or not.