Brar, Harnoor

Covid-19 Project

Reflection of the Project

This project was very real and close to both life and current situation, not only of a specific country, but the world as a whole. We may have been fighting with each other in the past, but this virus has brought us together and all the countries are working together with each other whether it is helping with supplies or joint researches on the vaccine. Working on this project has made me understand how the real-life projects are like.

Looking at the project, I thought I would do it with HashMaps, ArrayLists, and Priority Queues. HashMaps because this a big data, and I wanted the keys to automatically update value rather than me updating it everytime, this was best possible with HashMaps. I thought of ArrayList and Priority Queues to sort the values, but later I got rid of the Priority Queue because I could not find an easy way to print it in reverse order. For sorting in ArrayList, it was quite easier, I just called Collections.sort(arrayList) for sorting it in ascending order, and Collections.sort(arrayList, Collections.reverseOrder()) for printing the values in reverse. This was just my first look of how I would do it.

When I actually started doing the project, it seemed more complicated than I thought of it because the csv files had some typing errors, which I corrected. I made 3 hashMaps:

hMap : hash Map number 1, this map was created based on the “locations.csv” file. It was of type <String, String[]>, the key was “name of the country” because it is unique and it is present in both “locations.csv” and “full_data.csv”, so it seemed best option and it acted as a link between tow hMap1 and hMap2, and in the value, I created an array of string of size two and loaded the “continent of the country” at index [0], which would help me create hMap2 and the “population of the country” at index [1], which would later help me in hMap3 while printing the cases per million.

I decided to create an array of string as value of the hMap when I reached part 6, i.e. to print the cases per million. I did not want to create another data structure with another loop and load the population, so I created an array of string.

hMap2 : hashMap number 2, now this map created based on “hMap” (HashMap number 1) and “full_data.csv”. Now, this map was of type <String, Integer[]>, the key in this case was “continent”, because I had to print the number of cases and deaths according to continents. The value of hMap2 was an array of Integer of size two, where I loaded “totalCases” at index 0 and “totalDeaths” at index 2 of each continent.

I decided to create an array of integer because of the same reason that I listed in hMap number 1. I did not want to iterate over number of deaths and total cases differently and then loading them in different data structures, so I decided to create an array just once, and then keep adding the cases at index 0 and deaths at index 1.

hMap3 : HashMap number 3, this map is based on “full_data” and was of type <String, String>, the key for this map was “country”, and the value was “cases of each country”. The hMap3 and hMap1 would later help me in part 6 i.e. print cases per million.

With all these data structures I was able to follow along the project.

The main struggle that I encountered while doing this project was figuring out conditions for “if” and “else” statements. For me, this was the most difficult part, more than sorting. For example, the very first problem was ArrayOutOfBoundException while iterating through “locations.csv” because there were some lines which I had to delete or edit because of a typo. And in the "full_data.csv" there was some data of countries like "International" and "World", which I was not present in "locations.csv". So, figuring out the "if" part was quite mentally challenging for this problem. Also, other challenging conditions for "if" parts were like the conditions when it came to add data to the different hash maps.

What helped me to figure out these challenges were my pen and notebook. I wrote down the parts that were challenging and tried to explain to myself every single line and writing what output would it give. It helped to understand where the error was. For example, the error which troubled me most was the first line of "full_data.csv" which was "date, location……total_deaths". I didn't know it until I calmed myself and tried figuring out from my handwritten code on my notebook. All that I needed was to not read this line.

I did learn some new ways of reading through a file, how a real-life project looks like, and how to deal with errors. But the most important thing that I learnt was not to get frustrated, it's okay if the code didn't work as you expected. Being frustrated is making the things worse. All you need to do is take a break and go for a walk, or listen to music or whatever pleases you. In my case, new ideas came to my mind while walking on how to solve that error, and in some cases those ideas worked. Also, I learned a really cool in built in data type BigDecimal to preserve high decisions divisions. BufferedReader was also good to learn as I think it helped a lot in reading all the values of csv files. This project made me understand of how computer science helps to solve real life problems in just a matter of micro seconds. Moreover, I now have a clue of what coding in real life looks like and which I think will be very beneficial for me in upcoming future.

The parts where I had to print total number of cases and deaths in ascending and descending order. I, initially, thought of just creating one array list for ascending part and sort it. Then just print it from reverse for the “descending” part. But this would not work if I only call the method to print in descending order, because the list would not have been initialized. So, I had to create different lists for each part.

The part which I think could be improved is the case where I needed to print cases per million, I created a new Hash map in that method and passed “country” as key and the “cases per million” as value. For printing it in highest to lowest order, I initially created one more hash map and reversed the values, i.e. “cases per million” as keys and “country” as value. It worked fine, but the problem arose when “cases per million” were same for two countries, it overrode the other and only printed one. So, to get rid of this problem, I had to use comparator. I still think there should be an easy way to do this.