Final project MNXB01.

Daniel Nastasijevic, Yoana Ilarionova, Markus Müller

November 2018

Introduction.

The purpose of this project was to put all the knowledge acquired during the MNXB01 course to use in a all encompassing project covering the work in Linux terminal, working with git and github, implementing functions and classes with C++ as well as finalizing all the work and putting it in a presentable and easily readable way by using ROOT. In this particular project we gathered weather data from various weather stations all over Sweden that then proceeded to be analyzed in various ways such as the average temperature of each year, deviation from the average temperature, finding the hottest and coldest day of each year and how the latitude influences the temperature in any form.

Reading Data

The `readData()` function reads the data from the file and gets other important information such as the first recorded year and the number of years recorded. The function will start off by reading the header in all the data files except Uppsala (since they had a different data format) and then discard it since it is not useful information for the program to read in. Once this is done we start to actually read in the data properly with `input file stream`. Here the information we get will be presented in a two dimensional array where the first index is the year, the second index is the day of the year and the number stored is the temperature of that day that year. In the function the `startYear` is obtained by reading it out while at the beginning of the storing of the temperatures, as well as the number of fully recorded years.

Average Temperature.

The average temperature is calculated by taking in the sum of the temperature each day and then dividing by the number of days considerer which is done with a simple for loop.

In figure 1 one can see the the average temperature for every year for each weather station. The function works by taking the two dimensional array of the temperature per day per year, the number of years considered, the threshold for discarding elements and the array where the average

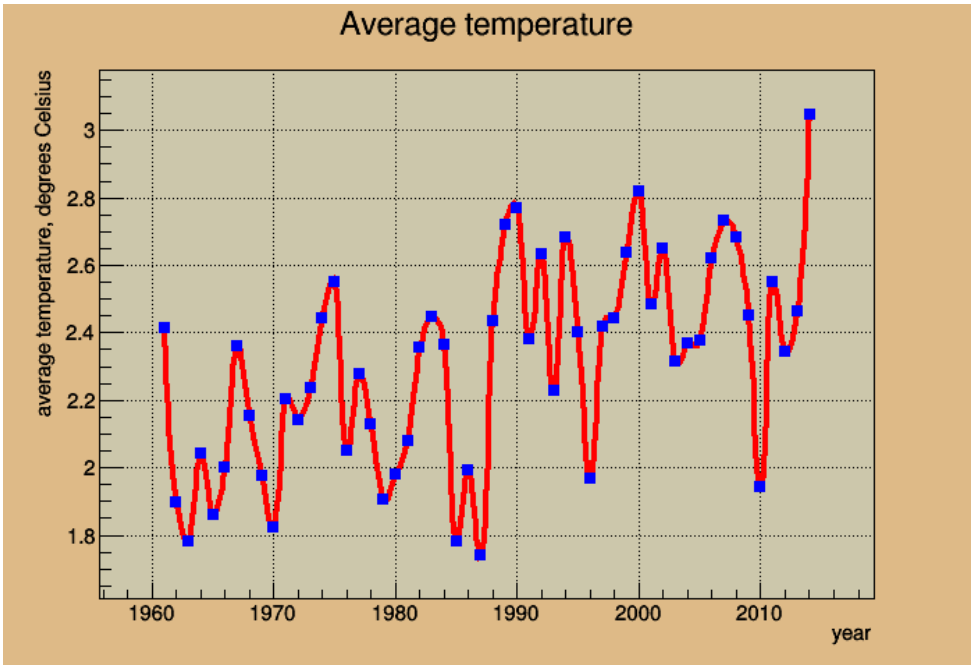

Figure 1: Average temperature for each year measured in Lund.

temperatures are going to be stored. It runs them through a `for` loop checking whether the temperature for a certain day is reliable, if its not it will be taken into account when calculating the average temperature. It will discard the unreliable values by a `if` statement that states if the element is smaller than the threshold it will be discarded (this is a feature of the reading).

Deviation from the overall average.

The deviation function takes the average that is acquired in the previous section and runs it through a `for` loop and subtracts it from the yearly average.

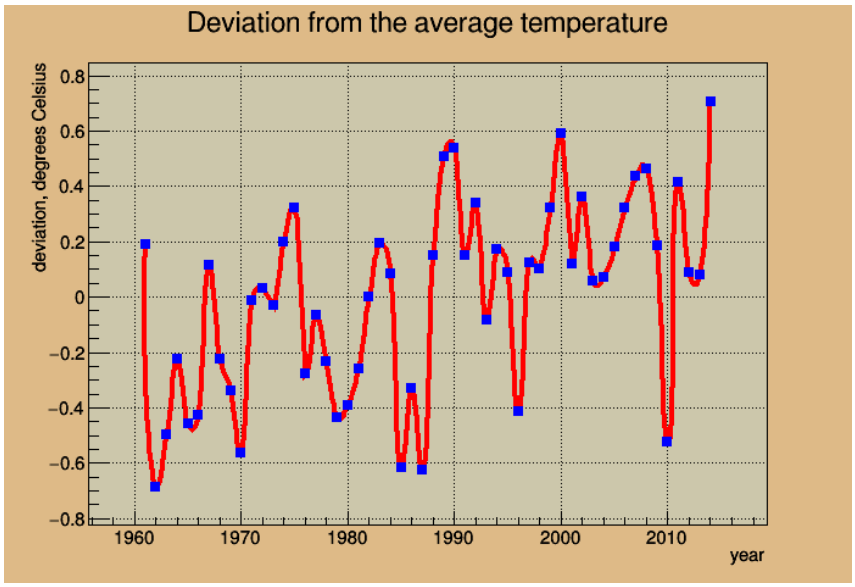

Figure 2: Deviation from the average temperature measured in Boras.

Screening of the dependance of the temperature on latitude.

After the values of all weather stations are averaged over all time, it is possible to have a plot over these averages against the latitude of where these measures were made in order to see if the difference in latitude is already a significant factor on the scale of a few degrees.
This was done by storing all average temperatures in an array. The values of latitude were read out of the files and also stored in an array of the same length. The array of latitude was sorted and then plotted against the array of temperatures.

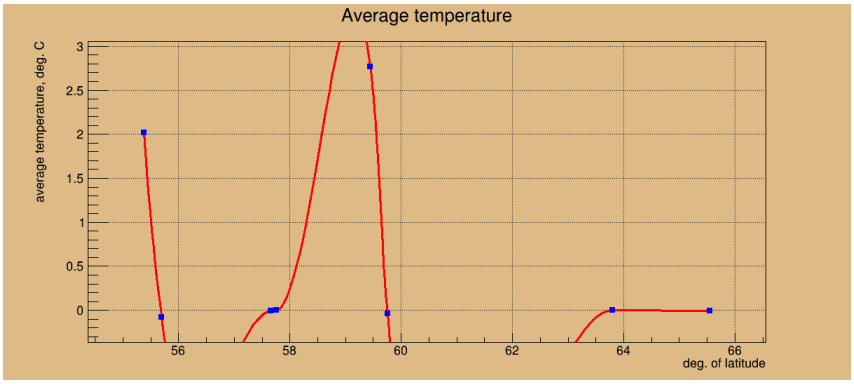

Figure 3: Result for plotting the overall average temperature of each weather station against their respective degree of latitude. There seems to be no significant correlation on these factors.

Hottest and coldest days of the year.

First, it has to be determined which day is the hottest and coldest day, respectively, for a given year.
For the hottest day, this was done with the C++ function `max_element`, which returns the pointer to the largest element of a certain range of addresses (one row of the temperature array in this case), and the function `distance`, which returns the difference between two addresses as an integer.
The determination of the coldest day had to be written in a more manual way since the set value of -275 used to flag datapoints as discarded would always be found as the lowest element of the row. The solution was to write a function that consists of a `for` loop over all days of the year. In this way, flagged measures can be disregarded as usual via an `if` clause. For all other valid days, a temporary variable will store the coldest day until a given time of the loop. The temperature of this day will be compared to the temperature of the currently considered day. If the latter is smaller than the former, the former will be overridden by the latter. This makes the temporary variable contain the day of the coldest temperature of a given year at the end of the loop.

By these means, one can call the functions just described in a loop that stores the respective result in two arrays for the hottest and coldest days of all given years. These can now be used to fill two histograms that count how often a given day was the hottest or coldest day of the year.

Conclusion.

After extended testing and corrections of the code one of the main and most important observations that were made was that where the data comes from and how the data is given from the stations part is just as important as how it is analyzed. The first issue with this was encountered when trying to analyze the data from Uppsala. Here the first problem was that this was the only data that had a header before everything with completely useless information to what we where actually looking for. The second problem from this data set was that the data that was listed was both incomplete and inconsistent. As in when we made a closer observation on it what became apparent was that Uppsala had periods in time where they simply did not take any data at all for years as well as that they had both gathered data from other stations as well as extrapolated said data. This led to an utterly unusable and unreadable fit once it was run through the program. The best way we found to circumvent this issue was to simply omit Uppsala's data all together.

A somewhat similar problem was found with the data from Falun weather station. Where in this case they did not operate at all during Christmas time as in from about 23rd of December all the way to 7th January. This in turn meant that it caused problems when reading the date.