STILL A DRAFT

MNXB01
Philip J. Fredholm
Lee Chun Hin

November 3, 2022

Contents

Introduction

In order to be able to draw conclusions about some physical phenomena, such as the climate of Earth, it is sometimes necessary to be able to process large amounts of data. In order to save a considerable amount of time doing so, it is often the case that such processing may be done with the help of a computer programming language. The purpose of the subject of this report has been to try to use the programming language C++ in order to to process data from the SMHI OpenData initiative and draw conclusion about climate and weather trends from this data. In particular, attempts have been made in order to analyse the data and find the distribution of coldest and hottest day of each year, the first day of summer as well as to analyse the temperature of a given day of the year (August 23rd) over many years.

Theory

In order to be able make the analysis of what day is the first day of summer, a definition of when summer starts must first be laid out. The definition chosen for this project was 5 consecutive days of an average temperature greater or equal than 10.0 °C.

In order to be able to read in the data to be analysed, a class called `readData` , in C++ code according to the C++17 standard, was created and used by placing a source file in a folder called `src` and a header file in a folder called `include` inside the folder called `PhilipCode` [1]. The class `readData` would take a file name for a file as the same folder as the final program and how many header lines of a text file with relevant (and properly formatted) data to read in. It would then have three getter-functions, one to get the first year of measurements, one to get the last year where the data had measurements, one to get the file name of the read in file and finally one to get the data that was read in. The read in data was stored in in a vector from the C++ standard library, which in turn contained one vector for each year that there had been measurements. Each vector for a year in turn contained twelve vectors, one for each year. Finally, each 'month-vector' contained 31 vectors, one for each possible day, within which (that is to say within the 'day-vectors') the average of the measured values for that day was stored. If no measurements were available for a given day, that 'day-vector' was left empty.

In order to actually find the warmest and coldest day of each year, as well as the start of summer, a file called `main.cpp` was placed in the folder `PhilipCode` and when compiled had access to the `dataReader` class by being compiled with a translation unit. This `main.cpp` file contained a function for finding either the hottest or coldest day of a given year by taking a pointer to a 'year-vector' of the type that was stored in the vector given by the getter function for the data of the `readData`-class. Similarly, there was also a function for finding the first day of summer, according to the definition presented earlier, by taking a pointer to a 'year-vector'. It should be noted that the way these functions operated was by creating a 'flattened' vector, which created a vector of as many elements as there were (nested) day vectors in the 'year-vector' given to the function as an argument. If a day had no measured values, it was still added to the 'flattened' vector and given a 'garbage' value, either extremely big or extremely small, as to not interfere with the logic used to find the desired day. It is important to note that since the 'year-vectors' month vectors always had 31 'day-vectors', the number of days in the flattened vector was always greater than 366, as even days which do not exist, such as a February 31st, were still added as a 'garbage' value. Finally, the `main.cpp` created output files storing the hottest and coldest days found of each year by the index of the hottest day in the flattened year vector. It also outputted a datafile with the index of the first day (in the flattened year vector) of the first day to meet the previously stated requirement for the start of summer.

In order to plot the results, three ROOT macros, called `hot.C`, `summer.C` and `summer.C`, and placed in the `PhilipCode` folder, were used. These macros read in the relevant data file created from the `main.cpp` file and created histograms with 366 bins from one to 366. It then added the number of counts from the read in processed data to the relevant bins. Note that for the macro `cold.C`, there was instead 400 bins between -200 and 199, and all counts for days with day number greater than 180 were given to the bin which had day number of 365 less than the relevant day. Note also that these 'days' give the day out of a 'year' where each of the twelve months have 31 days. The macros then also saved the histograms as graphs and printed the mean value of the day from the read in data and printed it to the standard output. Note that in [1] the macros, along with the processed data, have been moved to a folder called `rootmacros` in the folder `PhilipCode`.

Method

The `main.cpp` file was compiled using `g++` with an object file for the translation unit for the `readData`-class and the flags `-Wall`, `-Wextra` and `-Werror` and the version flag `-std=c++17`. The compilation was with the help of a program called `Make`. The outputted file was then ran, with the relevant data file to read from being placed in the same directory as the file which was ran, and data files containing the processed data were outputted. Following this, ROOT version 6.26/06 for Ubuntu 20 was ran on Ubuntu 20, running through Windows subsystem for Linux (WSL2). The macros `hot.C`, `cold.C` and `summer.C` were compiled with the `.L <filename>+`-syntax. The defined functions in these macros were then ran and the outputted values of the means were read off from the the standard output and the generated histograms were saved.

Results

The mean day as for being the coldest day of the year was found to be day 31.6812, the mean day for being the coldest day of the year was found to be day 192.428 and the mean day to be the start of summer was found to be day 139.051. The respective histograms for coldest and hottest days, as well as for the first day of summer, may be seen in Figure 1, Figure 2 and Figure 3 respectively.

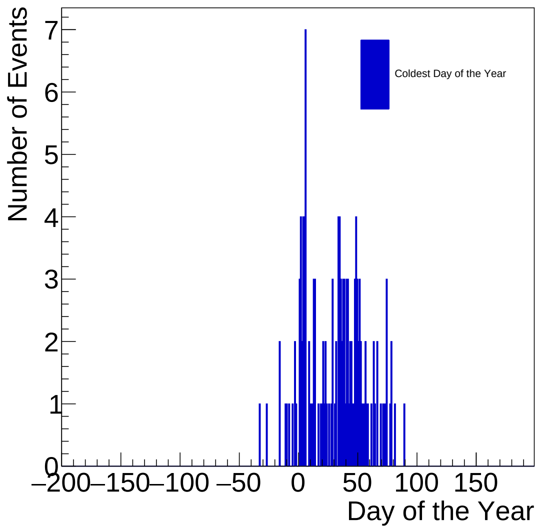


Figure 1: Shows the number of occurrences of a specific day being the coldest day of that year in Borås. Negative days indicate that the day was in the end of the previous year and its value for the day was shifted back 365 days. Note that the 'days' given here is the day out of a year where each of the twelve months are taken to have 31 days.

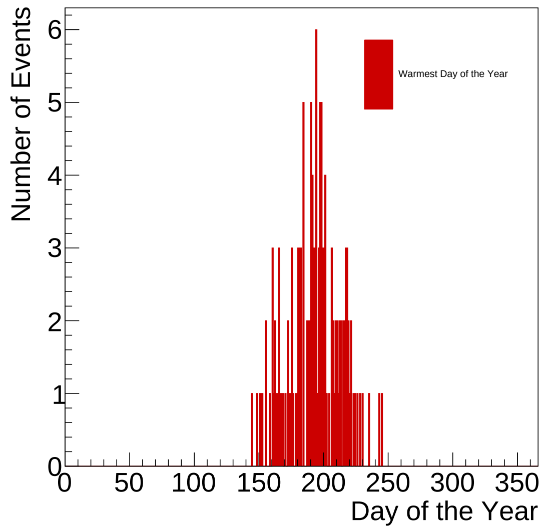


Figure 2: Shows the number of occurrences of a specific day being the warmest day of that year in Borås. Note that the 'days' given here is the day out of a year where each of the twelve months are taken to have 31 days.

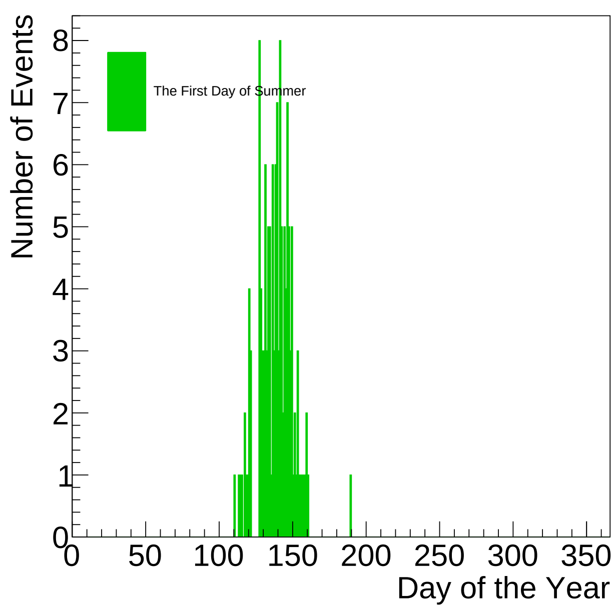
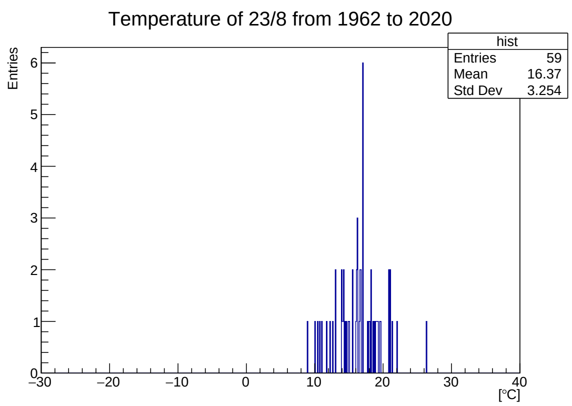
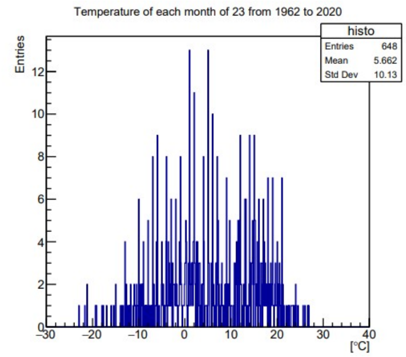


Figure 3: Shows the number of occurrences of a specific day being the first day of summer that year in Borås. Note that the 'days' given here is the day out of a year where each of the twelve months are taken to have 31 days.





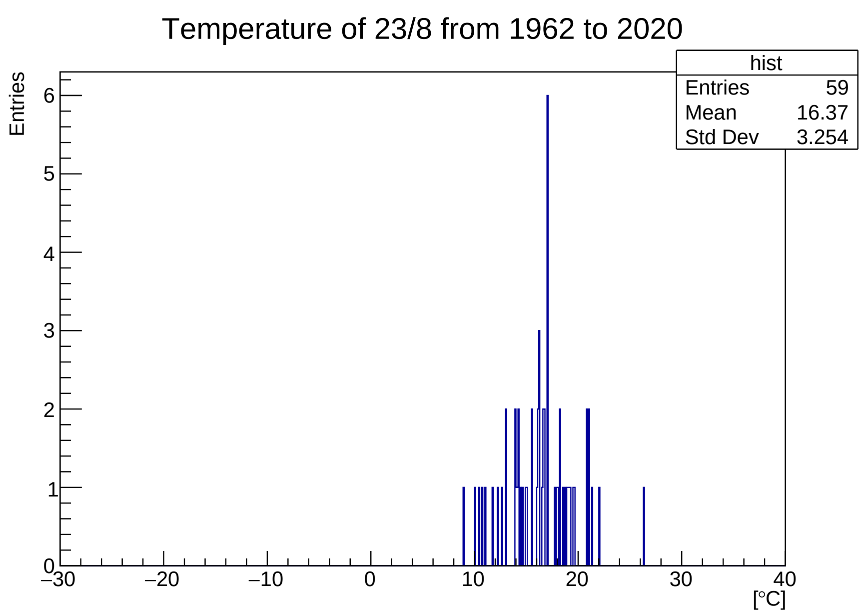


Figure 4: Temperature of Umeå Airport in 23/8 from 1962 to 2020.

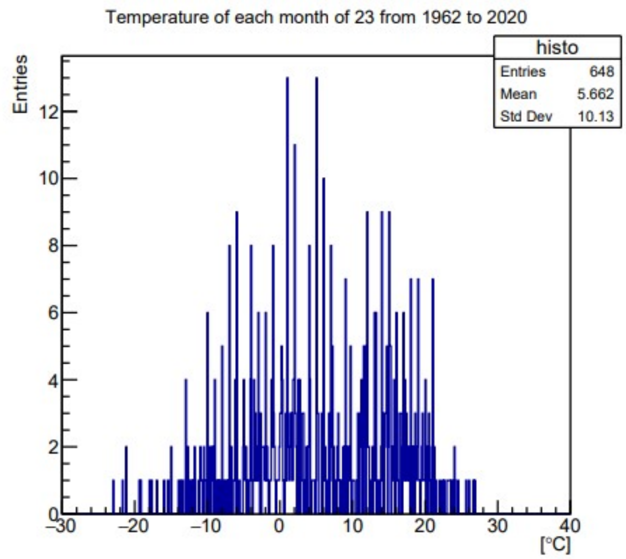


Figure 5: Temperature of Umeå Airport in each of the date of 23 since 1962 to 2020.

Discussion

It may be concluded that it was possible to produce histograms for the coldest and hottest days of the year, as well as for first day of summer. As for the calculated mean values for the relevant days, the coldest day was found to be, rounded upwards, to the 32nd day which would place it somewhere close to February 2nd, which seems reasonable. The first day of summer was found to occur around day 139. Considering that the start of June, the start of the six month where 5 months of about 30 days have passed, should be around day 150, this does not seem unreasonable. As for the hottest day of the year, the mean was around day 192. If the beginning of August occurs after seven months of approximately 30 days, that would place it at around day 210. Thus, from every day experience, a calculated value of 192 seems to be within reasonable bounds.

There are however some things which should be clearly remarked upon. That each month was assumed to have 31 days clearly means that the listed means do not corresponds to the actual day of an actual day of the Gregorian calendar. Secondly, as the days occurring after day 180 were shifted back by 365 days, while the used 'years' had 31 days each month, it is likely that many days for Figure 1 were not shifted properly. Since Figure 1 seems to mostly follow the normal distribution of a Gaussian, the large peak seemingly occurring 'off-center' of the thought of Gaussian could likely be explained by this. As for Figure 2 and Figure 3, these seem to, at least by eye, look somewhat like normal distribution. Furthermore, a significant improvement to taking the mean day for each respective quantity, it would probably had been better to fit Gaussians to the histograms in Figure 1, Figure 2 and Figure 3 and look for what value these were centred around.

As for other things that may be improved upon, using a separate library for both reading in the data and for storing the dates, as to make sure that the processed data actually refers to the Gregorian calendar, would probably have been a big improvement.

To conclude; it is clear that the programming tools were successfully able to identify climate trends and give qualitative conclusions. However, the conclusions would have benefited from a more thorough implementation in order to be able to draw more quantitative conclusions, especially regarding properly implementing the Gregorian calendar.

References

[1] The code in the folder "PhilipCode", available on a remote Git repository on GitHub at https://github.com/fredholmP/MNXB01-FinalProject .

[2] The code in the folders INSERT CHRIS'S FOLDERS LOCATION HERE available on a remote Git repository on GitHub at https://github.com/fredholmP/MNXB01-FinalProject .

[3] The data available from January 1st 1884 at 07.00 up until June 1st 2021 at 06.00 for the city Borås, provided by Lund Univeristy in association with the course MNXB01 during the autumn semester of 2022. At the time of writing, this data may be downloaded from the website https://www.smhi.se/data/meteorologi/ladda-ner-meteorologiska-observationer .

[4] The data available from DAY TIME up until DAY TIME for the "Umeå Flygplats" measuring station, provided by Lund Univeristy in association with the course MNXB01 during the autumn semester of 2022. At the time of writing, this data may be downloaded from the website https://www.smhi.se/data/meteorologi/ladda-ner-meteorologiska-observationer .