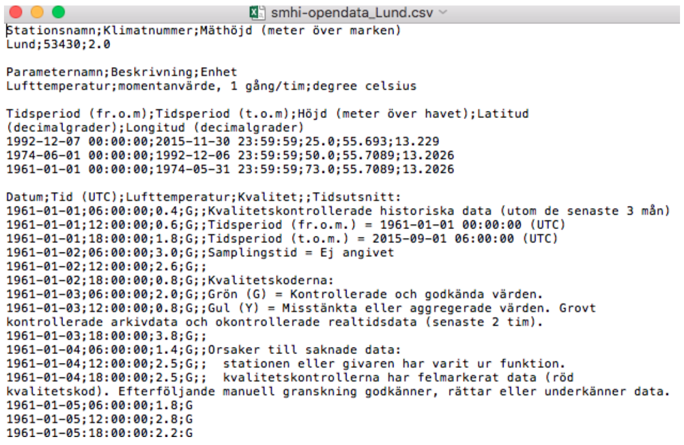
Report for final project, MNXB01

by Philip Siemund

2019-11-06

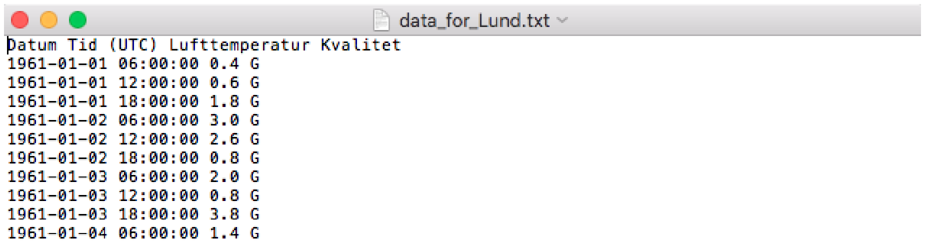
The goal of the project was to analyze data from the Swedish Meteorological and Hydrological Institute (SMHI). The data consisted of air temperature measurements from different towns in Sweden. The file format of the files containing the data was CSV. Figure 1 shows a screenshot of the file associated with Lund.



```
Stationsnamn;Klimatnummer;Mäthöjd (meter över marken)
Lund;53430;2.0

Parameternamn;Beskrivning;Enhet
Lufttemperatur;momentanvärde, 1 gång/tim;degree celsius

Tidsperiod (fr.o.m);Tidsperiod (t.o.m);Höjd (meter över havet);Latitud
(decimalgrader);Longitud (decimalgrader)
1992-12-07 00:00:00;2015-11-30 23:59:59;25.0;55.693;13.229
1974-06-01 00:00:00;1992-12-06 23:59:59;50.0;55.7089;13.2026
1961-01-01 00:00:00;1974-05-31 23:59:59;73.0;55.7089;13.2026

Datum;Tid (UTC);Lufttemperatur;Kvalitet;;Tidsutsnitt:
1961-01-01;06:00:00;0.4;G;;Kvalitetskontrollerade historiska data (utom de senaste 3 mån)
1961-01-01;12:00:00;0.6;G;;Tidsperiod (fr.o.m.) = 1961-01-01 00:00:00 (UTC)
1961-01-01;18:00:00;1.8;G;;Tidsperiod (t.o.m.) = 2015-09-01 06:00:00 (UTC)
1961-01-02;06:00:00;3.0;G;;Samplingstid = Ej angivet
1961-01-02;12:00:00;2.6;G;;
1961-01-02;18:00:00;0.8;G;;Kvalitetskoderna:
1961-01-03;06:00:00;2.0;G;;Grön (G) = Kontrollerade och godkända värden.
1961-01-03;12:00:00;0.8;G;;Gul (Y) = Misstänkta eller aggregerade värden. Grovt
kontrollerade arkivdata och okontrollerade realtidsdata (senaste 2 tim).
1961-01-03;18:00:00;3.8;G;;
1961-01-04;06:00:00;1.4;G;;Orsaker till saknade data:
1961-01-04;12:00:00;2.5;G;;  stationen eller givaren har varit ur funktion.
1961-01-04;18:00:00;2.5;G;;  kvalitetskontrollerna har felmarkerat data (röd
kvalitetskod). Efterföljande manuell granskning godkänner, rättar eller underkänner data.
1961-01-05;06:00:00;1.8;G
1961-01-05;12:00:00;2.8;G
1961-01-05:18:00:00:2.2:G
```

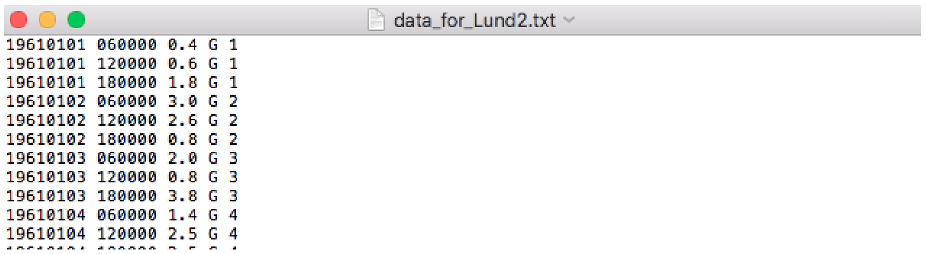
Figure 1: The raw data stored in a CSV file.

To extract only the measurements of the file, that is, all that follows after the line starting with "Datum; ..." and only the first four columns (using the field separator ";" for the definition of a column), a bash script tempdata.sh was written and executed. This script would extract the relevant data and save it to a new file data_for_town.txt. The script went through a couple of drafts, yielding different outputs as shown in figure 2.



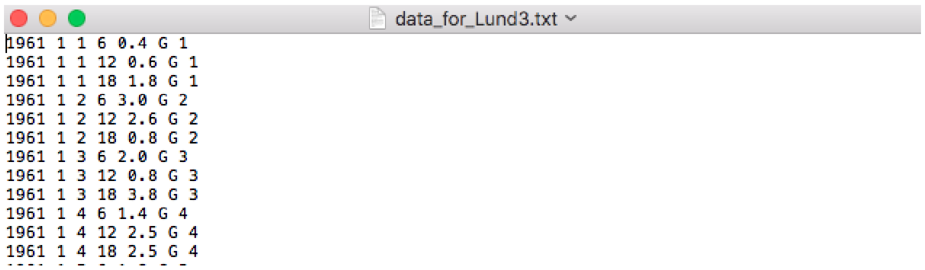
```
Datum Tid (UTC) Lufttemperatur Kvalitet
1961-01-01 06:00:00 0.4 G
1961-01-01 12:00:00 0.6 G
1961-01-01 18:00:00 1.8 G
1961-01-02 06:00:00 3.0 G
1961-01-02 12:00:00 2.6 G
1961-01-02 18:00:00 0.8 G
1961-01-03 06:00:00 2.0 G
1961-01-03 12:00:00 0.8 G
1961-01-03 18:00:00 3.8 G
1961-01-04 06:00:00 1.4 G
```

(a)



```
19610101 060000 0.4 G 1
19610101 120000 0.6 G 1
19610101 180000 1.8 G 1
19610102 060000 3.0 G 2
19610102 120000 2.6 G 2
19610102 180000 0.8 G 2
19610103 060000 2.0 G 3
19610103 120000 0.8 G 3
19610103 180000 3.8 G 3
19610104 060000 1.4 G 4
19610104 120000 2.5 G 4
```

(b)

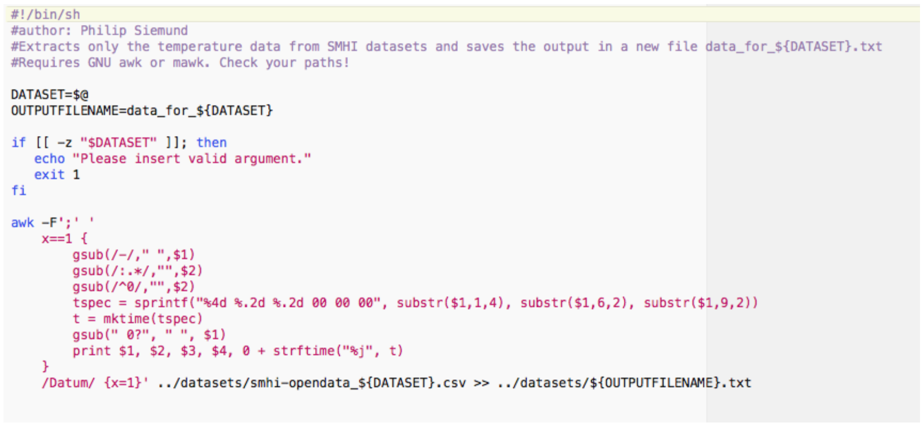


```
1961 1 1 6 0.4 G 1
1961 1 1 12 0.6 G 1
1961 1 1 18 1.8 G 1
1961 1 2 6 3.0 G 2
1961 1 2 12 2.6 G 2
1961 1 2 18 0.8 G 2
1961 1 3 6 2.0 G 3
1961 1 3 12 0.8 G 3
1961 1 3 18 3.8 G 3
1961 1 4 6 1.4 G 4
1961 1 4 12 2.5 G 4
1961 1 4 18 2.5 G 4
```

(c)

Figure 2: Different outputs of tempdata.sh; from earlier drafts, 2a and 2b, and final draft, 2c.

The last column in 2b and 2c specifies the day number of the year, i.e. a number between 1 and 365/366. This was achieved by creating a UNIX timestamp in tempdata.sh, which, by the way, used the single command GNU awk to format the data (see figure 3).

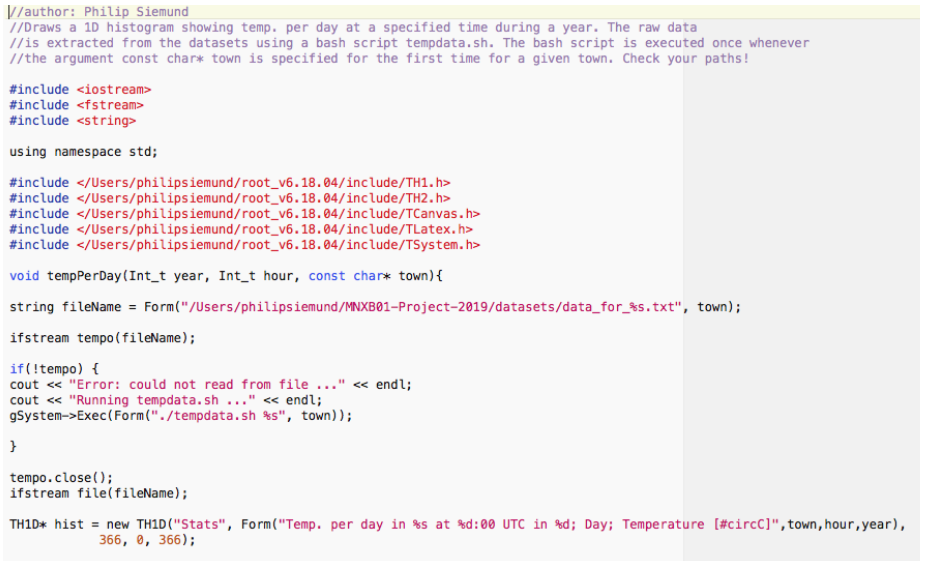


```
#!/bin/sh
#author: Philip Siemund
#Extracts only the temperature data from SMHI datasets and saves the output in a new file data_for_${DATASET}.txt
#Requires GNU awk or mawk. Check your paths!

DATASET=$@
OUTPUTFILENAME=data_for_${DATASET}

if [[ -z "$DATASET" ]]; then
   echo "Please insert valid argument."
   exit 1
fi

awk -F';' '
    x==1 {
        gsub(/-/," ",$1)
        gsub(/:.*/,"",$2)
        gsub(/^0/,"",$2)
        tspec = sprintf("%4d %.2d %.2d 00 00 00", substr($1,1,4), substr($1,6,2), substr($1,9,2))
        t = mktime(tspec)
        gsub(" 0?", " ", $1)
        print $1, $2, $3, $4, 0 + strftime("%j", t)
    }
    /Datum/ {x=1}' ../datasets/smhi-opendata_${DATASET}.csv >> ../datasets/${OUTPUTFILENAME}.txt
```

Figure 3: tempdata.sh.

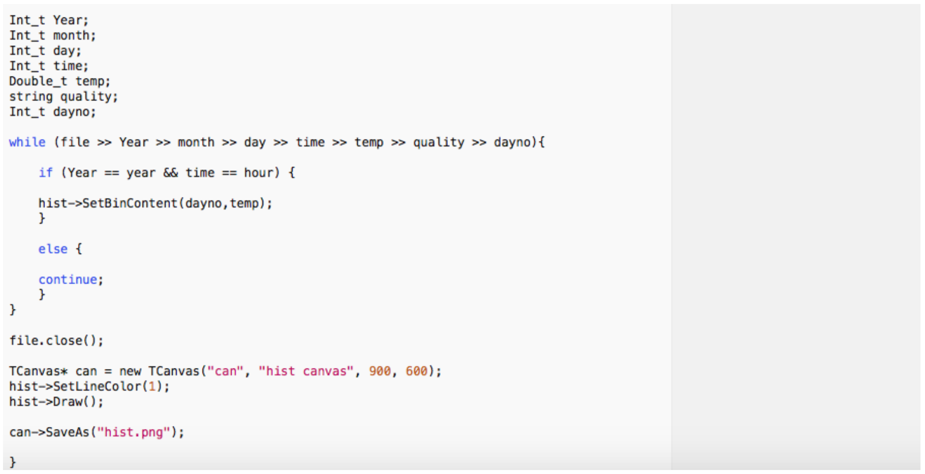
The purpose behind tempdata.sh was to make the data more accessible. Incuding the header fstream in a given C file, one could simply stream the different datatypes in the file data_for_town.txt onto corresponding variables in a while-loop. The data of interest could be specified by including conditional statements in the while-loop.

The file analyze_data.C exemplifies this (see figure 4). If one compiles the file in the data analysis tool ROOT and calls the function temPerDay(), it produces a histogram showing the temperature per day in a town at a given hour a given year. Some such histograms are shown in figure 5.



```
//author: Philip Siemund
//Draws a 1D histogram showing temp. per day at a specified time during a year. The raw data
//is extracted from the datasets using a bash script tempdata.sh. The bash script is executed once whenever
//the argument const char* town is specified for the first time for a given town. Check your paths!

#include <iostream>
#include <fstream>
#include <string>

using namespace std;

#include </Users/philipsiemund/root_v6.18.04/include/TH1.h>
#include </Users/philipsiemund/root_v6.18.04/include/TH2.h>
#include </Users/philipsiemund/root_v6.18.04/include/TCanvas.h>
#include </Users/philipsiemund/root_v6.18.04/include/TLatex.h>
#include </Users/philipsiemund/root_v6.18.04/include/TSystem.h>

void tempPerDay(Int_t year, Int_t hour, const char* town){

string fileName = Form("/Users/philipsiemund/MNXB01-Project-2019/datasets/data_for_%s.txt", town);

ifstream tempo(fileName);

if(!tempo) {
cout << "Error: could not read from file ..." << endl;
cout << "Running tempdata.sh ..." << endl;
gSystem->Exec(Form("./tempdata.sh %s", town));

}

tempo.close();
ifstream file(fileName);

TH1D* hist = new TH1D("Stats", Form("Temp. per day in %s at %d:00 UTC in %d; Day; Temperature [#circC]",town,hour,year),
            366, 0, 366);
```

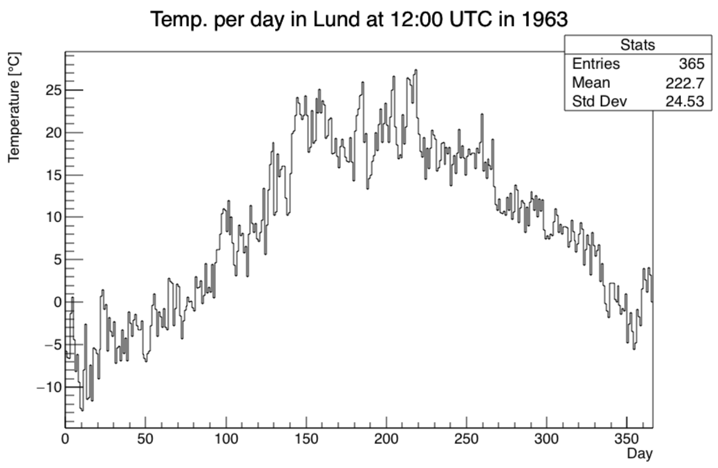
(a)



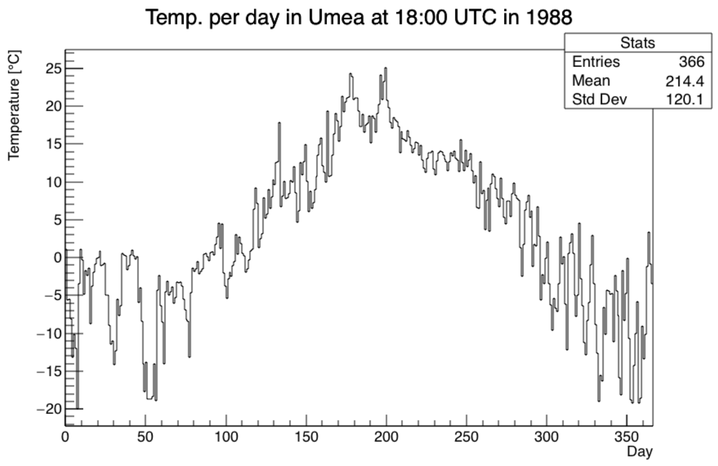
```
Int_t Year;
Int_t month;
Int_t day;
Int_t time;
Double_t temp;
string quality;
Int_t dayno;

while (file >> Year >> month >> day >> time >> temp >> quality >> dayno){

    if (Year == year && time == hour) {

    hist->SetBinContent(dayno,temp);
    }

    else {

    continue;
    }
}

file.close();

TCanvas* can = new TCanvas("can", "hist canvas", 900, 600);
hist->SetLineColor(1);
hist->Draw();

can->SaveAs("hist.png");

}
```

(b)

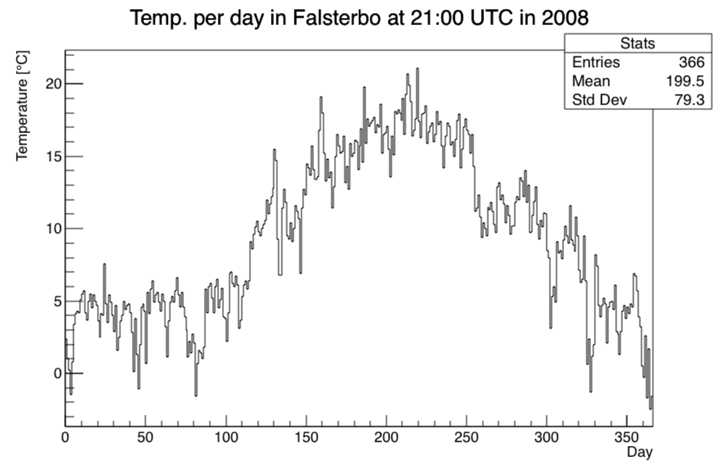
Figure 4: analyze_data.C.



(a)



(b)



(ƃ)

Figure 5: Example output of analyze_data.C.