

Project 1: Reporting Temperature Sensor Data

Posted: Wednesday February 14, 2007.
Described: Thursday February 15, 2007.
Due: 11:59PM, Monday March 6, 2007.

Project Objective:

1. get familiar with the process of completing a programming project.
2. explore different ways (table and figure) to report data.
3. learn to declare and use variables of the basic data types.
4. use array and basic loops for data manipulation.
5. understand the usage of command line arguments.
6. get familiar with how to read from and write into a file.

Project Description:

5 wireless sensors are deployed in a pond, from the surface to the bottom of the pond, to measure water temperature during the day. The sensors send their measured temperature information (in Fahrenheit) to a base station at the beginning of each hour. The base station collects all these data into a single text file (see the input section for format of this file).

Your program should read in this text file as input and report the data to a scientist who prefers to read the temperature in Celsius (see the output section for the detailed requirements of the report and output). A master program, in the format of executable file, and a sample input file will be provided at the class webpage.

Input

Your program will read in input from the data file generated at the base station. The file consists of multiple lines. Each line consists of exactly three fields representing the index of the sensor (1, 2, ..., 5); the time data is collected (0:00, 1:00, ..., 23:00); and the temperature (in Fahrenheit and ranges between 15F and 140F), respectively. For example,

```
1 0:00 35.024
3 0:00 39.782
2 0:00 37.540
4 0:00 33.901
5 0:00 32.658
2 1:00 37.166
```

...

temperature has 3 digits after the decimal point. These three fields are separated by exactly one single white space and there is no extra character at the end of each line.

Note that the first 5 lines are the temperature data at midnight (0:00), the next 5 lines are the data at 1:00 am, and so on. However, at each hour, the 5 data may not arrive in the same

order as their indices. For example, at midnight, sensor 3’s data arrives earlier than sensor 2; and at 1:00 am, sensor 2’s data is the first to arrive. This is caused by the wireless communication.

Output

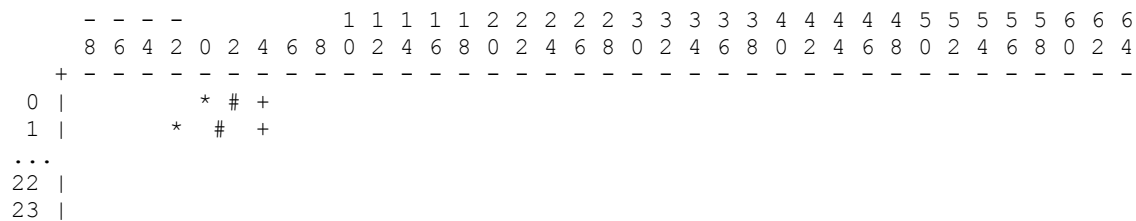
You program needs to generate 2 reports and write them to an output file as well as to the monitor. The third report is optional and you will receive bonus if you complete it correctly.

Report 1: Write all the temperature data from the input file to an output file of the following form

	sensor1	sensor2	sensor3	sensor4	sensor5	mean
00:00	1.680	3.078	4.323	1.056	0.366	2.101
01:00	...					
...						
23:00	...					
high	...					
low	...					
mean	...					

This table has 7 columns, the first column is 5-character wide indicating the time that data is collected; the other 6 columns are all 7-character wide for temperature data. The “mean” column gives the average temperature at a given time. There are exactly 2 white spaces between the columns. The “high/low/mean” rows give the highest/lowest/average temperatures of the day, respectively, reported by a sensor. All the temperature values are in Celsius and should have 3 digits after the decimal point (*hint: the formatted printf() statement will do the rounding*). A sensor’s measurement at any given time is from the input file (in Fahrenheit though), but you need to calculate the values for other entries.

Report 2: Depict the high/low/average temperature in the same plot as described below. Write it both to an output file and to the monitor.



The first two rows plot the indices along the x-axis, which is the temperature (in Celsius) and goes from -8C to 64C. The first row shows the negative sign and the number at the ten’s position, if exist. The second row shows the number at the one’s position. There are exactly 5 white spaces at the beginning of both rows. Only even number indices are printed. There is exactly 1 white space between them indicating the odd number indices (such as -7, -5, ...).

The third row starts with exactly 3 white spaces, followed by a “+” sign, and then the repeated pattern of “-” (1 white space and a negative sign), all the – signs are beneath the indices of the x-axis.

Project Template: (only for your reference, you do not have to use this)

```
// document your project and write down your name, student ID,
// section No. here

#include <stdio.h>

// define constants and functions if you are using them

int main(int argc, char *argv[])
{
    // declare the input and output files
    FILE* input, *output1, *output2;

    // declare other variables

    // some safety check
    if(argc !=4)
    { printf("Usage: executable inputfile output1 output2\n");
      exit(0);
    }

    // open the input and output files with more safety check
    input = fopen(argv[1],"r");
    if( input == NULL)
    { printf("File %s cannot open!\n", argv[1]);
    }

    Output1 = fopen(argv[2], "w");
    if( output1 == NULL)
    { printf("File %s cannot open!\n", argv[2]);
    }

    Output2 = fopen(argv[3], "w");
    if( output2 == NULL)
    { printf("File %s cannot open!\n", argv[3]);
    }

    // write your code here

    // close the input and output files
    fclose(input);
    fclose(output1);
    fclose(output2);

    return 0;
}
```