Programming Technical Report

Dr. Hoàng Đức Chính

|  |  |  |
| --- | --- | --- |
| Phạm Thị Trang  20181787 | Lê Minh Khuê  20181557 | Nguyễn Nam Anh  20181324 |

# **ABSTRACT**

# Report a software program (Programming language: C or C++) that uses functions and data structures to simulate a temperature sensor. The simulated sensor allows users to enter the measurement time, the measurement period and provide temperature values at a specific time, maximum value, minimum value, average value for each day. In addition, the sensor filters invalid values and sorts them in descending order.

# **INTRODUCTION**

# By using C++ programming language with functions and libraries, we simulated a temperature sensor. It has several main tasks such as: measuring temperature, filtering values, providing maximum - minimum values, sorting values in descending order and providing average temperature for the day. All output data is saved in .csv file. This report includes 6 parts explanation of ideas, programming design, algorithm flow, implementation, results, and assignment of duties.

# **EXPLANATION OF IDEAS**

## Task 1

## Simulating the sensor for any temperature in the range -25C to 125C at specified time. Then, it is saved in a \*csv file with subject line includes " Time & temperature values" for each measurement point.++ Task 1 allows the user to choose to default or import the measurement time and sampling cycle . Next, using functions in fstream library to work with the file, srand function to creat a random temparature values . The current time value is localtime, the measurement time entered is Setday, sampling period is Settime. Build the loop so that each loop print 1 line of values to the \*csv file and decrease Setday by 1 and it ends when Time <=0.

## Task 2

* 1. Task 2.1

Task 2.1 uses data from \*csv file created from task 1 and filter invalid values 150C−420C. Output data is saved in another \*csv file. Task 2 uses functions in vector library to separate the date, time and temperature value. Then, save the invalid in another vector and print each element of the vector to a new \*csv file.

* 1. Task 2.2

Task: Determine the maximum, minimum and average temperature values for each day . Save the results in a csv file Task 2.2 uses data from \*csv file created from task 2.1 and functions to separate date, time, and temperature values. For each day, find the maximum, minimum, and average values of the temperature and saved in a new \*csv file.

2.3 Task 2.3

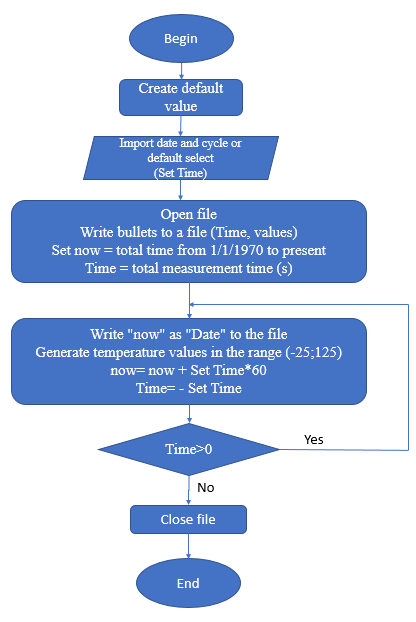
Task 2.3 count the number of temperature values in ranges and save the results in a new \*csv file. Task 2.3 uses data from \*csv file created from task 2.1 and separates time, date and temperature value. Store the satisfactory temprature values in different vectors. Use a counter variable and a loop to count the number of satisfactory values for each temperature range. Finally, save results in another \*csv file.

* 1. Task 2.4

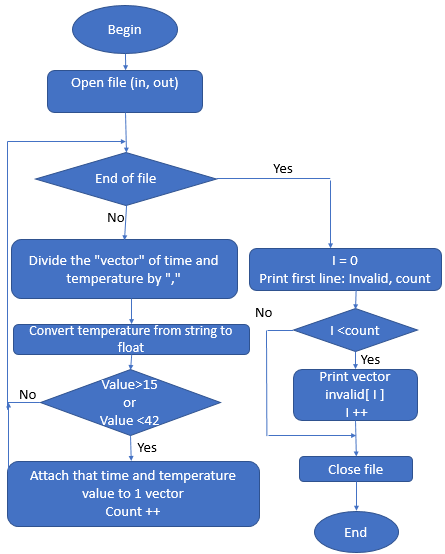
Task 2.4 sort the data in the file created from task 1 in descending order of temperature values. Task 2.4 uses data from \*csv file created from task 1 and separates time & date and temperature value in 2 different arrays. Next, create a loop find a maximum temperature value and then print the measurement time and that maximum temperature value in a new file. In the next loop, remove the maximum temprature value found in the previous loop from data. The loop ends when it read all data in the task 1

# **ALGORITHM FLOWCHART**

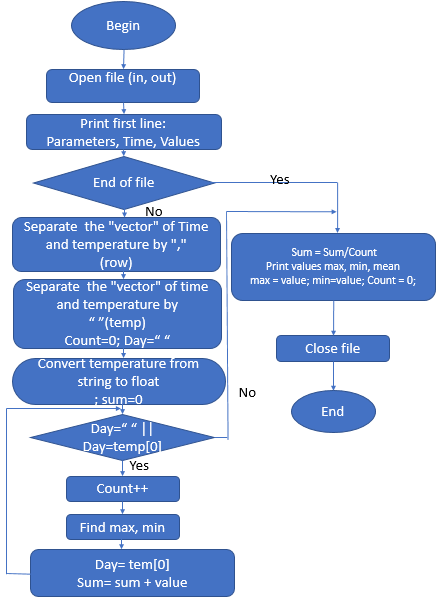
1. Task 1



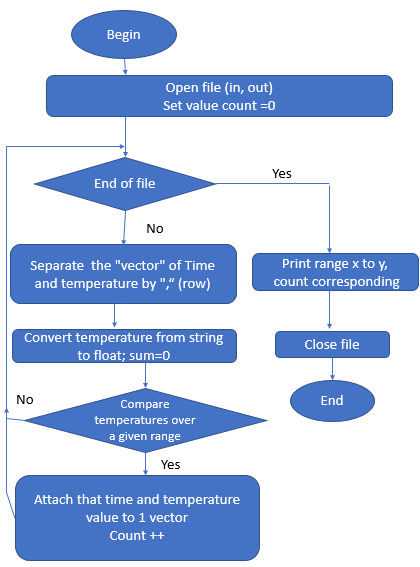
1. Task 2.1



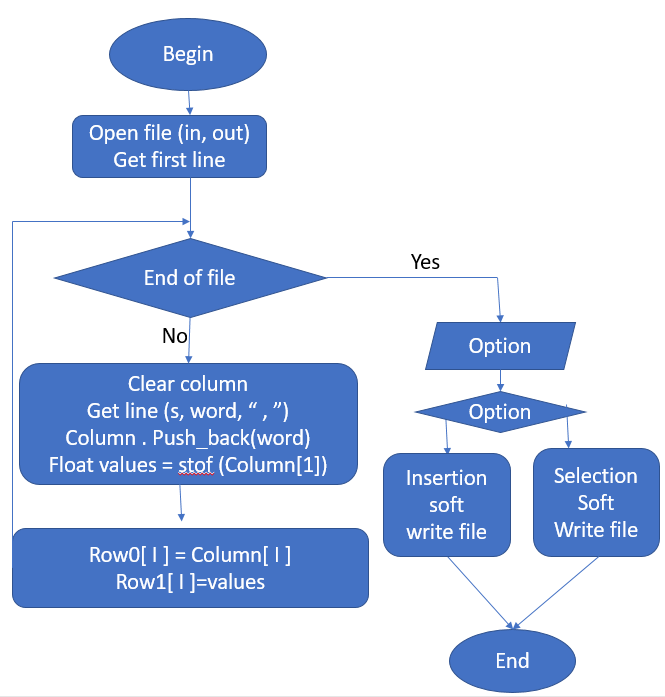
1. Task 2.2

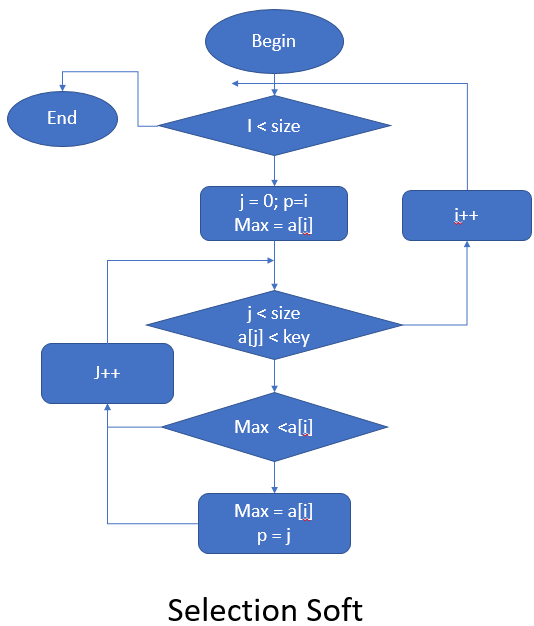
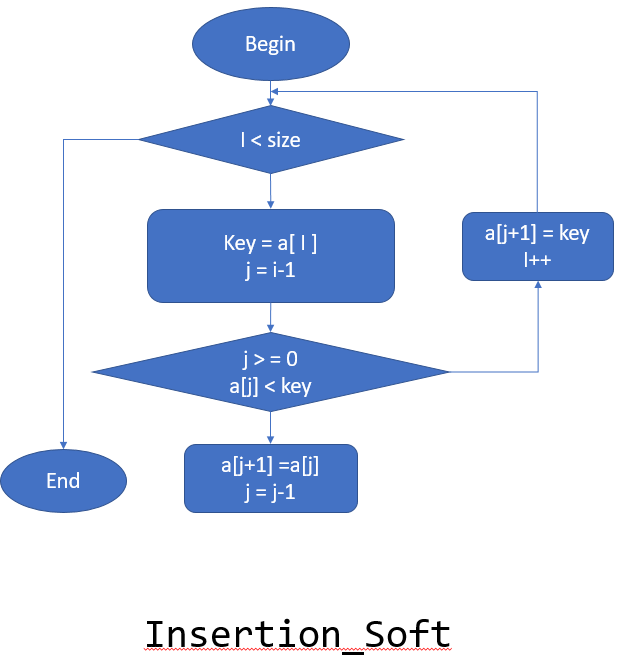


1. Task 2.3



1. Task 2.4





# **DESIGN PROGRAM**

1. Create data

Data include **Time** and **Values.** Data time point equidistant. Values random from -25 to 125. Resolution is 0.2.

+**Values:** Function **rand()** create random data but this data doesn’t change each time runcode. Function **srand(time(NULL))** helps **rand()** create random data change each time runcode.

+**Time:** using struct **time\_t** and function **localtime** get Pointer address to variable **now**, return numbers of second since 1/1/1970. Struct **tm** return Year, Month, Day, Hour, Minute and Second from 1/1/1900 of variable **now**

**+Loop:**  Number of loop is number of data point from extraction cycle and time measurement. Each loop create new random data and Time is **now+** **extraction cycle\*60**

1. Read data

Data file include **Subject line**, **Time** and **Values.**

Using class **Vector**, it’s similar to array with some some function but dosen’t need to declare size.

+**Read data.** Function **Getline(<file>,line)** read 1 line from file until new line. Data read save to string variable **line**. Function **getline(s, word, ',')** seprate data into **Time** and **Values** by char “**,**”

+ **Save data: <vector>.pushback** create new vecto similar to an element of array. Time and Data each line be saved in vector **Col0** and **Col1.**

**+Loop read data:**  Function **<file>.eof** check end of file. Read each line and save Time and Data

1. Oulier data

+Using Time and Values vector

+If Values under 15 or 42, save this values in Outlier Time and values vector, this value will be delete from original vector.

1. Summary data

+ Process each vector created from oulier data. Values is sorted by time next to each other. Separate Time to Date and time with function **getline(s,word,' ')**

+ Compare current day with previous day.

If they same, compare value with max and min current. If value under than min or higher than max, this value be saved to max or min

If they different, write Date, max, min and everage to file.

1. Static data

+ Using vector data right from Outlier data, compare value with range, and count number of values each range

1. Soft data

+ Give 2 option method sort data

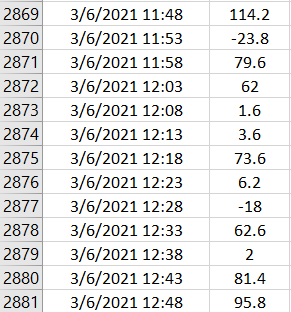
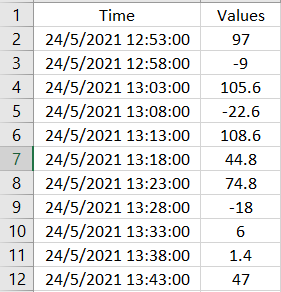
+ Insertion soft: Compare value with each value sorted and put each value into right position.

+ Selection soft: find max value in data unsorted and save to vector.

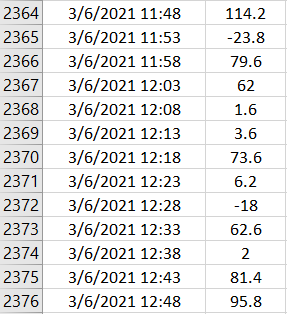
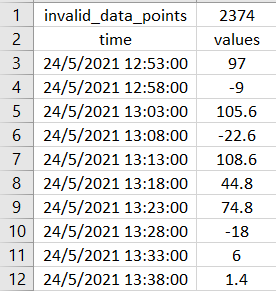
# **RESULTS AND MEMBERSHIP DUTIES**

1. Results

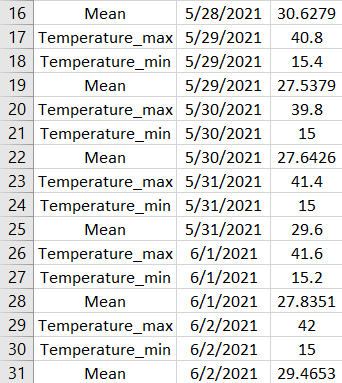
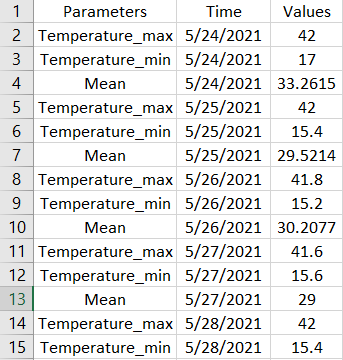
Task 1



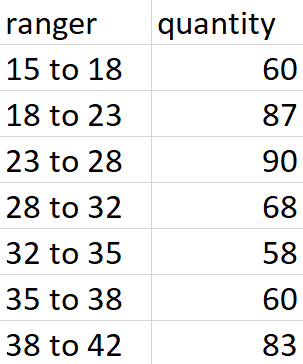
Task 2.1



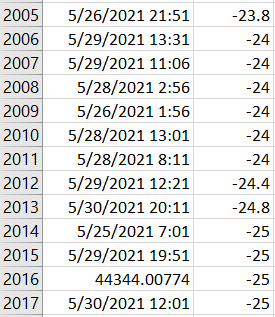
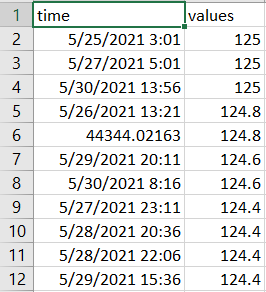
Task 2.2



Task 2.3



Task 2.4



# Membership duties

The whole group completed this great exercise together, the team members supported each other in the process. The specific duties of each member are as follows:

1. Lê Minh Khuê 20181557

In charge of brainstorming ideas throughout the problem. Implement task 2.3 and 2.4 programming.

1. Phạm Thị Trang 20181787

In charge of finding functions, optimizing algorithms and performing programming tasks 2.1 and 2.2.

1. Nguyễn Nam Anh 20181324

Program task 1, check the results and outline the report.