

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМ. Р.Е. АЛЕКСЕЕВА

ИНСТИТУТ РАДИОЭЛЕКТРОНИКИ И ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ

Кафедра «Информационные радиосистемы»

КУРСОВАЯ РАБОТА

*«Задан набор многократных измерений некоторой величины
несколькими датчиками. Описание отдельного измерения содержит
серийный номер датчика и числовое значение величины. Сформировать
набор разброса значений (минимальное и максимальное) по каждому из
датчиков.»*

Выполнил:

студент группы **24-Рз И. И. Иванов**

Проверил:

доцент кафедры ИРС С. Б. Сидоров

Нижний Новгород

2025

Содержание

Введение.....	3
1. Постановка задачи.....	5
2. Руководство пользователя.....	6
3. Руководство программиста.....	9
3.1 Структура программы.....	9
3.2 Структуры данных.....	10
3.3 Алгоритм вычисления минимального и максимального значения.....	11
Заключение.....	12
Список литературы.....	13

Введение

В работе рассматривается решение задачи разработки прикладной программы обработки множества многократных измерений некоторой физической величины, полученных от нескольких датчиков. Каждое отдельное измерение характеризуется серийным номером датчика и числовым значением измеряемой величины. Целью разработки является формирование набора разброса значений — минимального и максимального показаний — для каждого из датчиков.

Одним из способов упрощения решения данной задачи является использование отдельных функций, отвечающей за агрегацию и анализ данных. Такой подход позволяет декомпозировать задачу на ряд последовательных, более простых операций. Применение модульного подхода уменьшает затраты на отладку и повышает надёжность конечной программы. Важным аспектом также является использование структурированных типов данных для корректного отображения измерений и идентификаторов датчиков в программной модели.

В начале основной части отчёта приводится точная формализованная постановка задачи с указанием полного набора операций, которые должна обеспечивать прикладная программа: ввод данных измерений, группировка их по датчикам и вычисление минимальных и максимальных значений для каждого из них.

В руководстве пользователя раскрывается назначение программы, её функциональные возможности и реализуемые операции. Подробно описываются правила работы: способы ввода исходных данных, формат результатов и примеры взаимодействия пользователя с программой.

В руководстве программиста рассматриваются вопросы внутренней организации программы, включая перечень функций и схему их взаимодействия. Кроме того, описываются используемые структуры данных, а также наиболее

важные и интересные алгоритмы обработки — такие как группировка значений по датчикам и вычисление диапазонов измерений.

В заключении формулируются выводы о полноте решения поставленной задачи и приводятся возможные направления развития программы, такие как расширение форматов данных или добавление статистических показателей. Также включена техническая информация, содержащая листинги программного кода.

1. Постановка задачи

Рассматривается модель информации о многократных измерениях, получаемых от нескольких датчиков. Каждое измерение представлено совокупностью свойств: серийным номером датчика и числовым значением измеряемой величины. В области программной реализации модель отдельного измерения имеет вид структурного типа данных.

Рассматривается набор измерений с конечным количеством элементов. Требуется получить программную реализацию заданной обработки такого набора структурированных данных.

Все данные, необходимые для обработки, запрашиваются у пользователя: значения элементов, входящих в набор измерений, и дополнительная информация, необходимая для корректного выполнения обработки.

Обработка должна быть реализована отдельной функцией. При этом вся необходимая информация для выполнения обработки должна передаваться в функцию через список аргументов. Результат обработки набора структурированных данных — в данном случае минимальное и максимальное значение измеряемой величины для каждого датчика — также должен передаваться из функции через список аргументов.

Полученные результаты обработки должны быть выведены на монитор, то есть на стандартное устройство вывода.

Задан набор многократных измерений от нескольких датчиков. Требуется сформировать по каждому датчику диапазон разброса значений, определив минимальное и максимальное измеренные значения.

2. Руководство пользователя

Программа предназначена для обработки набора многократных измерений, полученных от нескольких датчиков, в режиме диалога с пользователем.

Программа позволяет пользователю задать количество датчиков, ввести измеренные значения для каждого датчика, после чего автоматически вычисляет минимальное и максимальное значение для каждого из них.

Программа является интерактивным консольным приложением. Весь диалог с пользователем происходит в текстовом режиме. Результаты обработки выводятся на экран монитора.

Запуск программы осуществляется либо вводом в командной строке имени исполняемого файла программы, полученного в результате компиляции, с последующим нажатием клавиши Enter, либо иным способом, зависящим от используемой операционной системы.

После запуска программы на экране появляется краткое информационное сообщение:

Sensor min/max program!

После информационного сообщения пользователю предлагается указать количество датчиков, для которых будут вводиться показания:

How many Sensors do we have?

После ввода количества датчиков, для каждого датчика последовательно выводится запрос на ввод его данных. Пример диалога:

Input serial number for [Sensor 1]:

Пользователь должен ввести серийный номер датчика.

Затем программа запрашивает ввод измерений для данного датчика:

Input values for [Sensor 1]:

[1]Value

Enter value:

Каждое значение вводится по одному. После ввода очередного значения пользователю задаётся вопрос:

Add another (y/n)?

- При вводе *y* пользователю предлагается ввести следующее измерение.
- При вводе *n* ввод значений для текущего датчика завершается.

Представленный процесс повторяется для каждого датчика до тех пор, пока не будут введены данные для всего набора.

После завершения ввода всех измерений программа автоматически выполняет обработку, для каждого датчика определяется минимальное и максимальное значение среди введённых измерений.

После завершения обработки на экран выводятся результаты в следующем виде:

Min/Max results per sensor:

Sensor 1: min: 12 max: 87 (serial: 101)

Sensor 2: min: -5 max: 42 (serial: 205)

Таким образом каждого датчика отображается:

- порядковый номер датчика (начиная с 1),

- минимальное значение,
- максимальное значение,
- серийный номер датчика.

Если для какого-либо датчика значения не были введены, выводится сообщение:

(no readings)

После завершения вывода результатов программа завершает свою работу.

3. Руководство программиста

3.1 Структура программы

Прикладная программа разработана с использованием принципов императивного программирования. Она представляет собой совокупность взаимодействующих функций. Структура программы отражена на рис. 1.



Рисунок 1: Структура программы

Программа состоит из следующих функций, назначение которых приведено ниже:

1. *main* — основная программа приложения;
2. *addSensor* — функция ввода измерений для одного датчика;
3. *findMinMaxAll* — функция определения минимальных и максимальных значений для каждого датчика;
4. *printMinMaxResults* — функция вывода полученных результатов на экран;
5. *repeat* — вспомогательная функция для интерактивного запроса о продолжении ввода данных.

3.2 Структуры данных

Для описания отдельного измерения определён структурный тип данных `Sensor`.

Из постановки задачи следует, что одно измерение характеризуется двумя величинами, каждому из этих свойств соответствует отдельное поле структуры:

- серийным номером датчика,
- числовым значением измеренной величины.

Определение структурного типа данных имеет следующий вид:

```
typedef struct Sensor {  
  
    int serial;  
  
    int value;  
  
} Sensor;
```

Для представления набора измерений используется вектор векторов:

```
std::vector<std::vector<Sensor>> TotalSensors;
```

- Вектор первого уровня соответствует набору датчиков.
- Вектор второго уровня содержит последовательность измерений одного датчика.

Для хранения результатов обработки используется отдельный структурный тип данных `MinMaxResult`, содержащий минимальное и максимальное значения, а также серийный номер датчика:

```
typedef struct MinMaxResult {  
  
    bool hasValues;  
  
    int sensorSerial;  
  
    int minValue;
```

```
int maxValue;  
} MinMaxResult;
```

Результаты по всем датчикам представляются в виде массива структур:

```
std::vector<MinMaxResult> MinMaxResults;
```

3.3 Алгоритм вычисления минимального и максимального значения

Определение диапазона (разброса) измерений по каждому датчику сводится к поиску минимального и максимального значений среди всех его измерений.

Алгоритм имеет следующую последовательность действий:

1. Для каждого датчика просматривается набор его измерений.
2. Если измерений нет, для данного датчика устанавливается флаг *hasValues = false*.
3. В противном случае:
 - значения *minValue* и *maxValue* инициализируются первым измерением;
 - далее последовательность показаний просматривается последовательно;
 - каждое значение сравнивается с текущими *min/max*:
 - если значение меньше текущего *min* — обновляется *minValue*;
 - если значение больше текущего *max* — обновляется *maxValue*.
4. По завершении обработки каждого датчика результаты заносятся в массив *MinMaxResults*.

Заключение

В данной работе задача разработки прикладной программы обработки набора измерений, выполняемых несколькими датчиками, была решена с использованием принципов императивного программирования. Программа осуществляет ввод данных от пользователя, группировку измерений по датчикам и определение минимального и максимального значения для каждого датчика.

На основании проведённой отладки и испытаний с использованием контрольных примеров можно сделать вывод, что разработанная прикладная программа корректно и в полном объёме решает поставленную задачу — формирование диапазонов (разбросов) значений по каждому датчику.

Список литературы

1. Керниган, Брайан У., Ритчи, Деннис М. Язык программирования С, 2-е издание. [пер. с англ.] / Б.У. Керниган, Д.М. Ритчи – М.: Вильямс, 2007.
2. Павловская, Т.А. С/С++. Программирование на языке высокого уровня: учебник для ВУЗов / Т.А. Павловская. – СПб.: Питер, 2009.
3. Орлов, С.А. Технологии разработки программного обеспечения. учеб. пособие. 2-е изд./ С.А. Орлов, – СПб.: Питер, 2003. – 480 с.: ил.
4. Борисенко, В.В. Основы программирования / В.В.Борисенко, – Интернет-университет информационных технологий – ИНТУИТ.ру, 328 стр. – 2005 г.
5. Шилдт, Г. Полный справочник по С: учеб. пособие / Г. Шилдт. – 4-е изд. – М.: Изд. дом "Вильямс", 2008.
6. Костюкова, Н.И. Язык Си и особенности работы с ним / Н.И. Костюкова, Н.А. Калинина – Интернет-университет информационных технологий – ИНТУИТ.ру, 208 стр. – 2006 г.

Приложение А

Заголовочный файл – header.h

```
#ifndef VAR_36
#define VAR_36

#include <iostream>
#include <vector>

typedef struct Sensor {
    int serial;
    int value;
}Sensor;

std::vector<std::vector<Sensor>> TotalSensors;
Sensor dataToPush;

int currentWorkingSensor = 1;
int totalSensors;

typedef struct MinMaxResult {
    bool hasValues;
    int sensorSerial;
    int minValue;
    int maxValue;
} MinMaxResult;

std::vector<MinMaxResult> MinMaxResults;

int addSensor();
int findMinMaxAll();
void printMinMaxResults();

#endif //VAR_36
```

Основная программа – main.cpp

```
#include "header.h"

int repeat(){
    fflush(stdout);
    char answer;
    std::cout << "\nAdd another (y/n)?" << std::endl;
    std::cin >> answer;
    if (answer == 'y'){return true;}
    else{return false;};
}

int addSensor(int currentWorkingSensor){
    std::vector<Sensor> vectorToPush;
```

```

        std::cout << "Input serial number for [Sensor " << currentWorkingSensor
<<"]:" << std::endl;
        std::cin >> dataToPush.serial;

        std::cout << "Input values for [Sensor " << currentWorkingSensor << "]: " <<
std::endl;

        //debug
        int i = 1;

        while(true){

            //debug
            std::cout << "[" << i << "]" << "Value" << std::endl;

            std::cout << "Enter value: ";
            std::cin >> dataToPush.value;

            vectorToPush.push_back(dataToPush);

            //debug
            std::cout << "Current vector:\n";
            for (int i = 0; i < vectorToPush.size(); ++i){

                std::cout << "[" << i << "]" - " << "Serial: " <<
vectorToPush[i].serial << " Value: " << vectorToPush[i].value << std::endl;
            }

            if(!repeat()){
                std::cout << "End entering values for Sensor " <<
currentWorkingSensor << ";" << std::endl;
                break;
            }
            ++i;

        }

        TotalSensors.push_back(vectorToPush);

        return 0;
}

int findMinMaxAll(){

    MinMaxResults.clear();

    for (int i = 0; i < (int)TotalSensors.size(); ++i){
        MinMaxResult result;
        result.hasValues = false;

        const std::vector<Sensor> &sensorReadings = TotalSensors[i];

        if (sensorReadings.empty()){
            MinMaxResults.push_back(result);
            continue;
        }

        result.hasValues = true;
        result.minValue = sensorReadings[0].value;
        result.maxValue = sensorReadings[0].value;
    }
}

```

```

        result.sensorSerial = sensorReadings[0].serial;

        for (int j = 1; j < (int)sensorReadings.size(); ++j){
            if (sensorReadings[j].value < result.minValue){
                result.minValue = sensorReadings[j].value;
            }
            if (sensorReadings[j].value > result.maxValue){
                result.maxValue = sensorReadings[j].value;
            }
        }

        MinMaxResults.push_back(result);
    }

    return 0;
}

void printMinMaxResults(){
    if (MinMaxResults.empty()){
        std::cout << "No min/max results available. Run findMinMaxAll() first."
<< std::endl;
        return;
    }

    std::cout << "\nMin/Max results per sensor:" << std::endl;
    for (int i = 0; i < (int)MinMaxResults.size(); ++i){

        const MinMaxResult &result = MinMaxResults[i];
        std::cout << "Sensor " << i+1 << ": ";

        if (!result.hasValues){
            std::cout << "(no readings)" << std::endl;
            continue;
        }

        std::cout << "min: " << result.minValue << " max: " << result.maxValue;
        std::cout << " (serial: " << result.sensorSerial << ")" << std::endl;
    }
}

int main(){

    std::cout << "Sensor min/max program!" << std::endl;

    std::cout << "How many Sensors do we have?" << std::endl;
    std::cin >> totalSensors;

    //FILL SENSORS
    for(int i = 1; i <= totalSensors; ++i){

        addSensor(currentWorkingSensor);

        ++currentWorkingSensor;
    }

    //DEBUG
    for (int i = 0; i < TotalSensors.size(); ++i){

```

```

        std::cout << "\n--- Sensor " << i+1 << " ---" << std::endl;

        for (int j = 0; j < TotalSensors[i].size(); ++j){
            std::cout << "[" << i+1 << "] - " << "Serial: " << TotalSensors[i]
[j].serial << " Value: " << TotalSensors[i][j].value << std::endl;

            }
        }

        findMinMaxAll();
        printMinMaxResults();

        return 0;
}

```