Báo cáo dự án: Thiết kế chương trình xử lý dữ liệu cảm biến nồng độ bụi

Vũ Huy Hoàng   
Trường Điện – Điện tử  
Đại học Bách khoa Hà Nội  
Hà Nội, Việt Nam  
vuhuyhoang.hust@gmail.com

Tổng quan— Thiết kế chương trình phần mềm với các hàm và cấu trúc dữ liệu thích hợp để xử lý dữ liệu cảm biến.

Từ khoá—dust, time

# Mở đầu

Dự án yêu cầu viết chương trình sử dụng ngôn ngữ C hoặc C++ với các hàm và cấu trúc dữ liệu thích hợp để mô phỏng dữ liệu cảm biến bụi PM2.5 đo nồng độ hạt bụi có kích thước < 2.5 microns trong không khí.

Khoảng đo:

Độ phân giải:

Dự án bao gồm 3 nhiệm vụ:

* Task 1: Tạo file dữ liệu từ cảm biến;
* Task 2: Phân tích dữ liệu cảm biến;
* Task 3: Mã hoá dữ liệu để truyền tin.

Trong báo cáo này, tôi sử dụng ngôn ngữ C++.

# Task 1

## Ý tưởng chính

Cấu trúc thư mục:

Các thư viện sử dụng:

|  |  |
| --- | --- |
| * iostream * fstream * string.h | * cstdlib * ctime * iomanip |

## Thiết kế chi tiết

Task 1 sử dụng 4 hàm là: *check\_err* (kiểm tra lỗi nhập tham số), *assign* (xác định giá trị các biến *n, si, st*), *exp\_data* và *write\_err* để xuất dữ liệu ra file (dữ liệu sensor và thông báo lỗi).

### Hàm “exp\_data”

**void exp\_data(ofstream &myfile, int \*vari)**

trong đó, myfile là file cần xuất dữ liệu cảm biến ra;

vari là con trỏ tới mảng lưu trữ tham số:

vari[0]: số cảm biến;

vari[1]: thời gian trích mẫu;

vari[2]: thời gian đo.

Lưu đồ thuật toán:

Diagram

Description automatically generated

### Hàm “main”

**int main(int argc, char \*\*argv)**

trong đó, argc là số lượng tham số đầu vào;

argv là chứa các tham số đó.

Lưu đồ thuật toán:

Diagram

Description automatically generated

## Kết quả và đánh giá

Chương trình chạy đúng với các trường hợp đầu vào đầy đủ, thiếu một hoặc một vài (cặp) tham số.

Chương trình chưa xử lý được trường hợp các cặp tham số không đúng thứ tự.

# Task 2

## Ý tưởng chính

Cấu trúc thư mục:

Các thư viện sử dụng:

|  |  |
| --- | --- |
| * iostream * fstream * string.h * cmath | * cstdlib * ctime * iomanip * vector * cstring |

## Thiết kế chi tiết

Task 2 sử dụng 2 header file quan trọng là: *export\_lib.h* (xuất dữ liệu ra các file) và *check\_string.h* (kiểm tra tính chất của xâu). Ngoài ra, chương trình còn một hàm quan trọng khác là *import\_data.*

### Các hàm trong “check\_string.h”

bool isBlank(string str); kiểm tra xâu str là rỗng/chỉ chứa khoảng trắng

bool isPosIntNumber(string s); kiểm tra xâu str là một số nguyên dương

bool isPosRealNumber(string s); kiểm tra xâu str là một số thực dương

bool isTime(string s); kiểm tra xâu str có dạng thời gian: NNNN:TT:NN GG:PP:GG

Nhìn chung các hàm trên không khó để lập trình, nhưng quan trọng trong việc kiểm tra lỗi của dữ liệu đầu vào trong task 2 và task 3 sau này.

### Hàm “import\_data”

**void import\_data(vector<dust\_data> &data\_valid, vector<dust\_data> &data\_outlier, int &count\_valid, int &count\_outlier, ifstream &myfile, int &num\_sen, ofstream &error\_file);**

trong đó, data\_valid là vector các dữ liệu hợp lệ;

data\_outlier là vector dữ liệu không hợp lệ;

count\_valid là biến đếm dữ liệu hợp lệ;

count\_outlier là biến đếm dữ liệu không hợp lệ;

myfile là file dữ liệu đầu vào;

num\_sen là số sensor;

error\_file là file thông báo lỗi.

Lưu đồ thuật toán:

Diagram

Description automatically generated

### Hàm “main”

**int main(int argc, char \*\*argv)**

trong đó, argc là số lượng tham số đầu vào;

argv là chứa các tham số đó.

Lưu đồ thuật toán:

Diagram

Description automatically generated

## Kết quả và đánh giá

Chương trình chạy đúng với các file dữ liệu đầu vào khác nhau.

# Task 3

## Ý tưởng chính

Cấu trúc thư mục:

Các thư viện sử dụng:

|  |  |
| --- | --- |
| * iostream * fstream * string.h | * cstdlib * ctime * iomanip |

## Thiết kế chi tiết

Task 3 sử dụng header file “check\_string.h” giống ở task2. Ngoài ra có thêm các hàm khác: *check\_err\_dataline* (kiểm tra tính hợp lệ của các cột dữ liệu), *import\_data* (lưu trữ dữ liệu hợp lệ)

### Hàm “import\_data”

**void import\_data(int &count1, byte\_seq &temp, struct tm &tm, string word)**

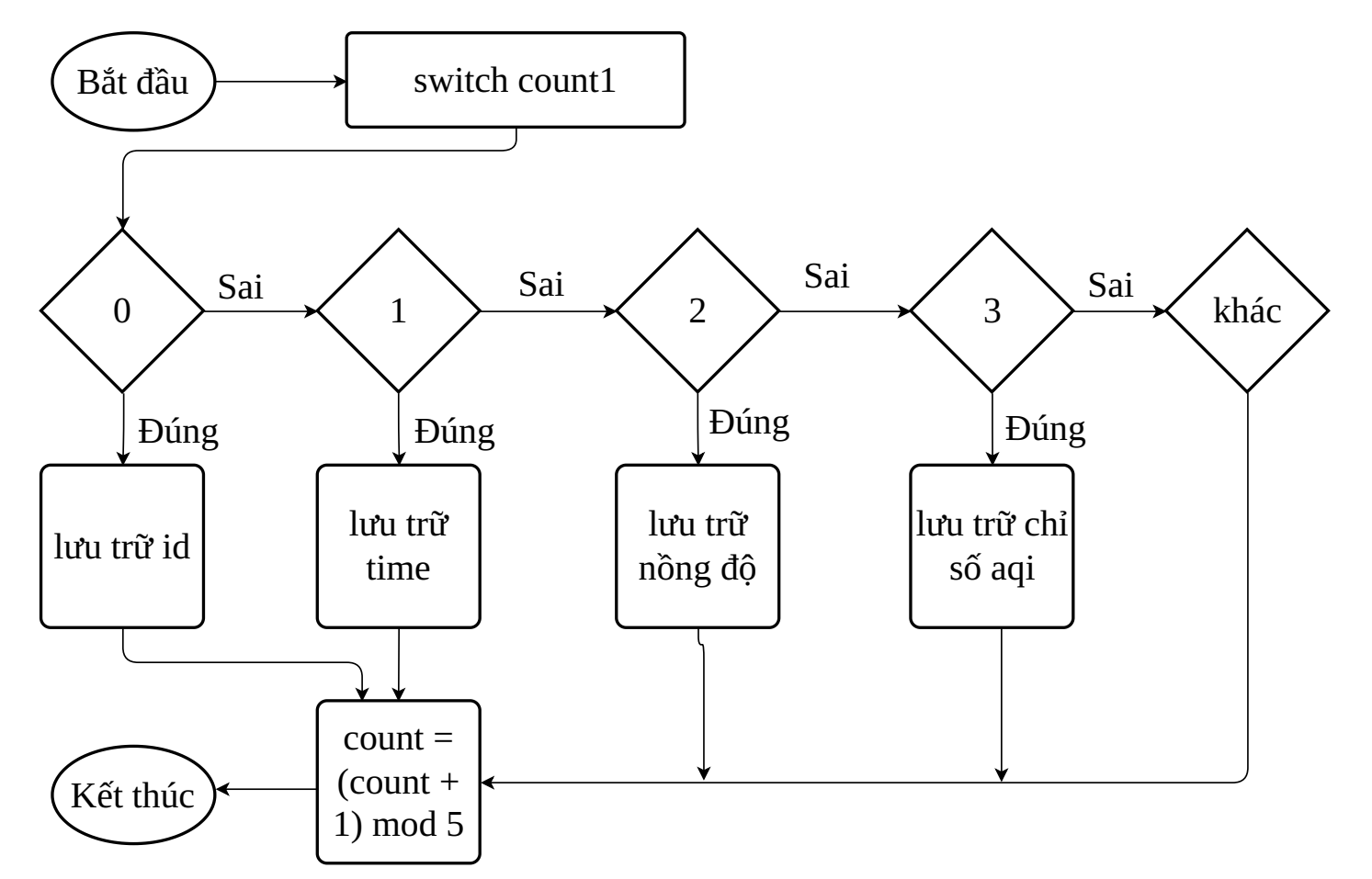
trong đó, count1 là chỉ số cột dữ liệu đang đọc;

temp là đối tượng có thuộc tính đọc từ input;

tm là biến lưu trữ thời gian;

word là xâu lưu trữ trường dữ liệu đang đọc;

Lưu đồ thuật toán:



### Hàm “main”

**int main(int argc, char \*\*argv)**

trong đó, argc là số lượng tham số đầu vào;

argv là chứa các tham số đó.

Lưu đồ thuật toán:

Diagram

Description automatically generated

## Kết quả và đánh giá

Chương trình chạy đúng với các file dữ liệu khác nhau.

# Tổng kết