

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN – ĐHQG_HCM
KHOA CÔNG NGHỆ THÔNG TIN



An toàn và phục hồi dữ liệu

Đồ án cuối kỳ

Giảng viên:

**Thái Hùng Văn
Ngô Đình Hy**

MSSV

Họ và tên

20120083

Nguyễn Trọng Hiếu

20120144

Lê Chí Nghĩa

Mục Lục

1.	Tiếp cận bài toán.....	3
2.	Cấu trúc của file	3
3.	Các yêu cầu phụ	4
4.	Phương thức thiết kế	5
4.1	global.h.....	7
4.2	fileTemplate.h	9
4.3	fileCombine.h.....	10
5.	Nhận xét.....	13

1. Tiếp cận bài toán

Thiết kế 1 định dạng file mới (file kết hợp) với cấu trúc đặc biệt để lưu trữ dữ liệu của 2 file con một cách an toàn với các yêu cầu phụ đi kèm.

2. Cấu trúc của file

Đặt vấn đề

- Vấn đề 1:** Dưới góc nhìn của một hệ thống lưu trữ tổng quát thì các kích thước của tập tin được lưu trữ phải có kích thước cố định. Tuy nhiên hai tập tin đưa vào có kích thước giãn nở được.
- Vấn đề 2:** Thỏa mãn được các yêu cầu phụ được nêu bên dưới.

Cấu trúc đề xuất

Cấu trúc chính

META DATA			FAT		DATA
COMBINE FILE	FILE 1	FILE 2	FILE 1	FILE 2	

Thiết kế phân vùng DATA

DATA				
CLUSTER 1 (8 SECTORS)	CLUSTER 2 (8 SECTORS)	CLUSTER N (8 SECTORS)
FILE 1 FILE 2 (7 SECTORS) / (1 SECTORS)

Khả năng của cấu trúc

❖ Giải quyết các vấn đề được đặt ra

- Vấn đề 1:** Với cấu trúc đề xuất, các thông tin được lưu trữ sẽ thông qua một text file có khả năng thay đổi kích thước tùy ý. Sau khi thông tin đã được lưu vào text file thì sẽ gọi cơ chế quản lý của cấu trúc lưu trữ text file đó vào trong cấu trúc. Cụ thể sẽ được nêu trong phần phương thức thiết kế.
- Vấn đề 2:** Đáp ứng các yêu cầu phụ theo các tiêu chí sau
 - Tránh việc dồn dữ liệu:** Dữ liệu sẽ vẫn có thể tiếp tục thêm vào khi không gian lưu trữ tương ứng của file đó của file kết hợp còn trống. Do đó không có khả năng bị dồn dữ liệu giữa 2 file.
 - Các phần tử đã xóa không xóa hẳn để có khả năng khôi phục lại:** Các phần tử đã xóa không hẳn đã xóa mà chỉ gán các giá trị trên cluster lưu trữ giữ liệu trong bảng fat của file tương ứng về trạng thái không còn được sử dụng. Khi đó với việc lưu trữ mở sẽ tìm kiếm các cluster với giá trị đại diện cho cluster trống (chưa từng bị xóa) để tiến hành lưu trữ mới. Khả năng phục hồi lại sẽ phụ thuộc vào thiết kế bổ sung trên phần metadata tương ứng của file trong cấu trúc chính.
 - Có thêm thông tin quản lý thiết lập:** Thông tin quản lý thiết lập có thể bổ sung trong phần metadata tương ứng với tập tin kết hợp cũng như từng tập tin con bên trong.

- **Mỗi cá nhân có tổ chức minh họa thông tin cơ bản:** Mỗi cá nhân được thiết kế với các thông tin gắn liền với nhau chẳng hạn như Struct myStruct. Thông tin đại diện cho cá nhân sẽ được nối tuần tự các giá trị bên trong struct và tiến hành mã hóa sau đó đẩy vào file tạm. Mỗi thông tin của 1 cá nhân chiếm 1 dòng trong file tạm.
- **Có cơ chế kiểm tra mật khẩu:** Mật khẩu được hỗ trợ cho cả file kết hợp cũng như từng file con bên trong. Hiện tại mật khẩu đang được lưu trữ dưới dạng cơ chế HashMD5 của password người dùng đưa vào.

❖ **Khả năng mở rộng**

- Tùy thuộc vào mục đích người dùng mà có thể thêm các thông tin quản lý tập tin con vào các phần metadata tương ứng.
- Đối với trường hợp không xóa hẳn dữ liệu, thiết kế được đưa ra nhằm mục đích giải quyết vấn đề này, tuy nhiên do thời hạn thực hiện đề án nên vẫn chưa thực hiện được mục đích trên.

3. Các yêu cầu phụ

- Tránh việc dồn dữ liệu
- Các phần tử được xóa sẽ không xóa hẳn để có khả năng phục hồi lại (ngoại trừ tình huống đặc biệt cần phải xóa hẳn, các phần tử đã xóa quá lâu cũng không cần phải phục hồi).
- Mỗi cá nhân có thể tổ chức minh họa vài thông tin cơ bản (Mã, Họ Tên, Ngày sinh, Ngày tham gia, Số ĐT, Số CCCD, ...), trong đó số CCCD và số ĐT cần bảo mật.
- Nên có thêm các thông tin quản lý cần thiết, như ngày tạo lập, thời điểm cập nhật, công thức /key mã hóa /giải mã, ...
- Cần có cơ chế kiểm tra mật khẩu động hoặc passkey mỗi khi file được mở, có không chế thời gian (nhập sai nhiều lần thì phải đợi một thời gian sau mới có thể nhập tiếp, vẫn sai nữa thì thời gian đợi bị tăng thêm).

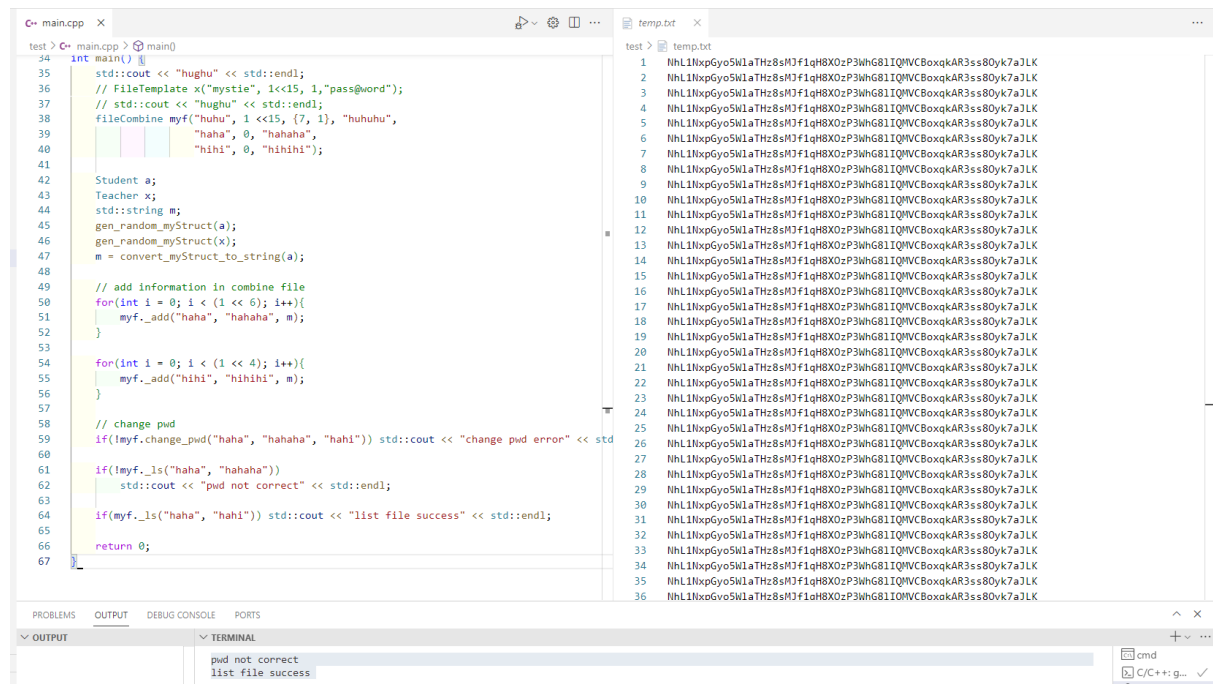
4. Phương thức thiết kế

Được cụ thể hóa thông qua các file sau:

final_lab

```
| -global.h           // định nghĩa về các thông tin dùng cho toàn bộ các file
| -fileTemplate.h     // tạo định dạng mẫu lưu trữ cho 1 file con
| -fileCombine.h       // tạo định dạng lưu trữ cho file combine
| -main.cpp            // chương trình demo final_lab
```

Kết quả demo của hàm main:



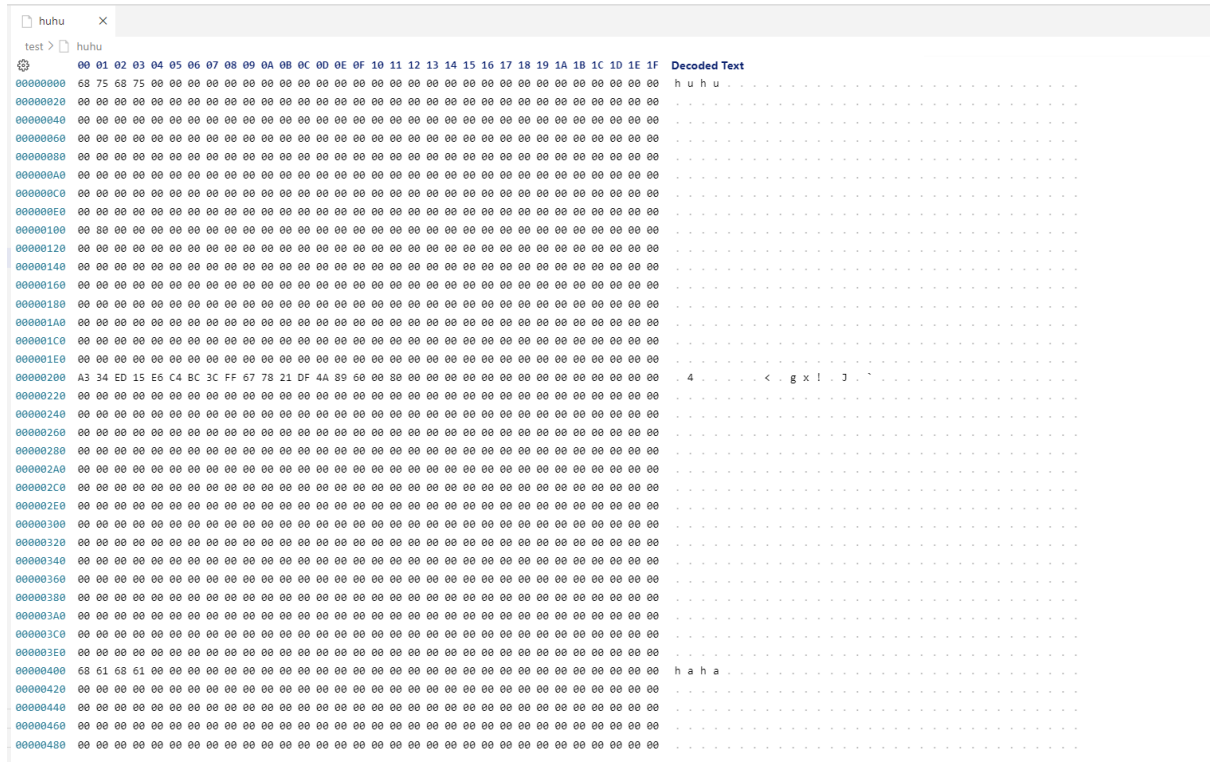
```
test > C++ main.cpp > [main]
35 int main() {
36     std::cout << "huhu" << std::endl;
37     // FileTemplate x("mystie", 1<15, 1, "pass@word");
38     // std::cout << "huhu" << std::endl;
39     fileCombine myf("huhu", 1 <15, {7, 1}, "huhuhu",
40                     "haha", 0, "hahaha",
41                     "hihi", 0, "hihihi");
42
43     Student a;
44     Teacher x;
45     std::string m;
46     gen_random_myStruct(a);
47     gen_random_myStruct(x);
48     m = convert_myStruct_to_string(a);
49
50     // add information in combine file
51     for(int i = 0; i < (1 << 6); i++){
52         myf._add("haha", "hahaha", m);
53     }
54
55     for(int i = 0; i < (1 << 4); i++){
56         myf._add("hihi", "hihihi", m);
57     }
58
59     // change pwd
60     if(!myf.change_pwd("haha", "hahaha", "hahi")) std::cout << "change pwd error" << std::endl;
61
62     if(!myf._ls("haha", "hahaha"))
63         std::cout << "pwd not correct" << std::endl;
64
65     if(myf._ls("haha", "hahi")) std::cout << "list file success" << std::endl;
66
67     return 0;
68 }
```

```
test > temp.txt
1 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
2 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
3 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
4 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
5 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
6 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
7 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
8 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
9 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
10 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
11 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
12 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
13 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
14 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
15 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
16 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
17 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
18 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
19 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
20 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
21 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
22 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
23 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
24 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
25 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
26 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
27 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
28 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
29 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
30 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
31 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
32 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
33 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
34 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
35 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
36 NhLiNxpgYo5WlaThz8sMjF1qH8X0zP3WhG81IQMVCBoxqkAR3ss80yk7aJLK
```

OUTPUT

```
pwd not correct
list file success
```

Hình 1: Chạy thử hàm main, một số thông tin liên quan được thể hiện trong phần [kỹ thuật làm việc với hệ thống](#)



Hình 2: File tổng hợp được thể hiện dưới dạng nhị phân

4.1 global.h

```
11 inline const int sector_size = 512;
12 inline const int cluster_size = 8;
13 inline const int n_sectors_for_file_info = 3;
14 inline const int n_sectors_for_combine_file_info = 8;
15
16 inline const std::vector<std::pair<std::string, unsigned int>> my_pair = {
17     {"end_of_file", 0xFFFFFFFF},
18     {"unallocated", 0x00000000},
19     {"erased", 0x00000001}
20 };
21 inline const std::vector<std::string> size_type = {"byte", "sector", "cluster"};
22
23 struct Size {
24     unsigned int bytes;
25     unsigned int sectors;
26     unsigned int clusters;
27 };
28
29 struct SizeStateForSector {
30     unsigned int total;
31     unsigned int empty_left; // unallocated
32     unsigned int real_empty_left; // unallocated + erased
33     unsigned int first_real_empty;
34 };
35
36 typedef struct{
37     std::string mssv; // 10bytes
38     std::string name; // 10bytes
39     std::string birth_date; // 10bytes
40     std::string join_date; // 10bytes
41     std::string phone; // 10bytes
42     std::string id_card; // 10bytes
43 }Student, Teacher, myStruct;
44
45 Size convert_to_Size(unsigned int& size, const std::string& type="byte");
46
```

Hình 3: Tổng quát của global.h

```

5 void gen_random_myStruct(myStruct& a) {
6     // Function to generate a random string of fixed length (10 bytes)
7     auto generate_random_string = []() -> std::string {
8         const std::string characters = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
9         std::random_device rd;
10        std::mt19937 gen(rd());
11        std::uniform_int_distribution<> dis(0, characters.size() - 1);
12
13        std::string result;
14        for (int i = 0; i < 10; ++i) {
15            result += characters[dis(gen)];
16        }
17
18        return result;
19    };
20
21    // Generate random values for each member of myStruct
22    a.mssv = generate_random_string();
23    a.name = generate_random_string();
24    a.birth_date = generate_random_string();
25    a.join_date = generate_random_string();
26    a.phone = generate_random_string();
27    a.id_card = generate_random_string();
28 }

```

Hình 4: Khởi tạo một đối tượng bất kì

4.2 fileTemplate.h

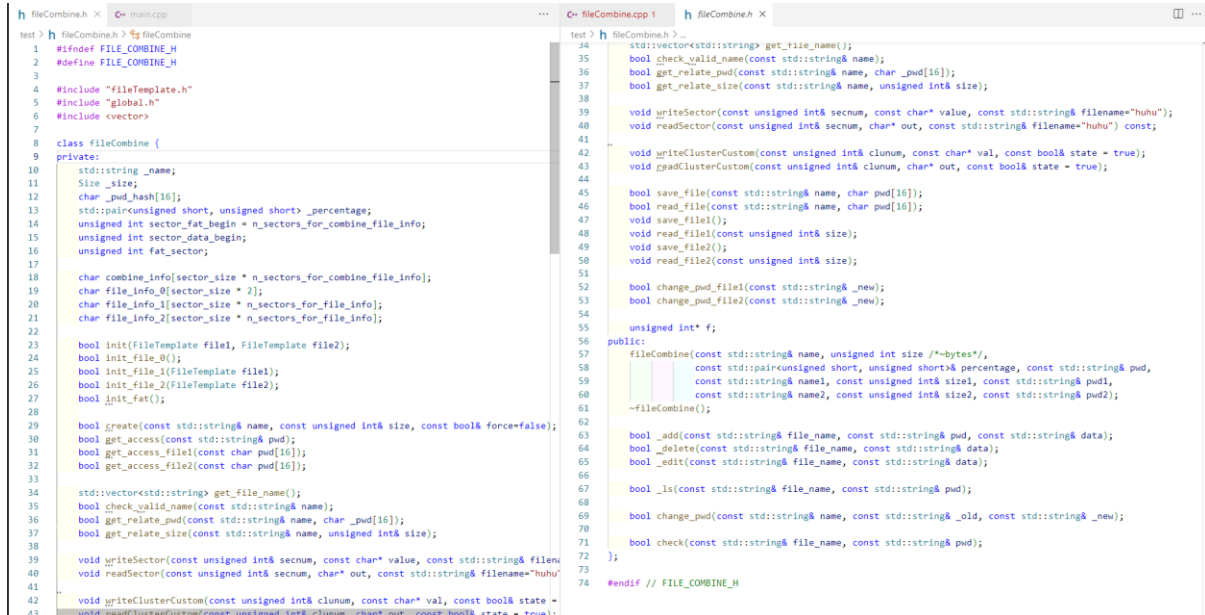
```
20 class FileTemplate {
21 private:
22     std::string _name;
23     Size _size;
24     bool _state;           // 0 - truy cập ít, 1 - truy cập nhiều
25     char _pwd_hash[16];    //
26
27     char template_infor[sector_size * n_sectors_for_file_infor];
28     char file_infor_0[sector_size]; // file_infor
29     char file_infor_1[sector_size]; // file_secure
30     char file_infor_2[sector_size]; // data_infor
31
32     bool init();
33     bool init_file_infor_0();
34     bool init_file_infor_1();
35     bool init_file_infor_2();
36
37 public:
38     FileTemplate(const std::string& name, unsigned int size /*~bytes*/,
39                 const bool& state, const std::string& pwd);
40     bool get_template_infor(char* output, size_t bytes=sector_size * n_sectors_for_file_infor);
41
42     bool openFile(const std::string& filename, const std::string& pwd);
43 };
```

Hình 5: Tổng quát của fileTemplate.h

Trong class template này được định nghĩa biến trạng thái state, khi state = true tương ứng với file sẽ chiếm kích thước lớn và thường xuyên được đọc ghi, ngược lại (state = false) thì file sẽ chiếm kích thước nhỏ và ít được truy suất hơn.

Các sector của metadata cho từng file còn trông tương đối nhiều, tùy thuộc vào mục đích người dùng có thể tiếp tục tận dụng để lưu trữ các thông tin liên quan.

4.3 fileCombine.h



Hình 6: Tổng quát của fileCombine.h

Kỹ thuật làm việc với hệ thống

Coi như hệ thống file combine như 1 chương trình quản lý và có pwd riêng. VD khi so sánh với hệ điều hành, mật khẩu để login vô user sử dụng tương ứng với mật khẩu để khởi tạo cũng như sử dụng của file kết hợp, mật khẩu của file con giống như mật khẩu của các tập tin lưu trữ bên trên màn hình máy tính.

1. Thêm – xóa – sửa

Đối với các lệnh thao tác với dữ liệu (thêm/xóa/sửa): thông qua đọc ghi bằng cluster – sector để thao tác. Do đó cần thiết lập cơ chế đọc cluster theo cấu trúc data được thiết lập bên trên (7 sectors đầu của cluster dành cho file 1 và 1 sector cuối của cluster dành cho file 2)

```

222 void fileCombine::writeClusterCustom(const unsigned int& clunum, const char* val, const bool& state){
223     unsigned int sz = 0;
224     if(state) sz = _percentage.first;
225     else sz = _percentage.second;
226     unsigned int sector_begin = clunum*cluster_size + sector_data_begin;
227
228     for(int i = 0; i < sz; ++i){
229         if(state)
230             this->writeSector(sector_begin+i, val + i*sector_size);
231         else
232             this->writeSector(sector_begin+i+_percentage.first, val + i*sector_size);
233     }
234 }
235
236 void fileCombine::readClusterCustom(const unsigned int& clunum, char* out, const bool& state){
237     unsigned int sz = 0;
238     if(state) sz = _percentage.first;
239     else sz = _percentage.second;
240     unsigned int sector_begin = clunum*cluster_size + sector_data_begin;
241
242     for(int i = 0; i < sz; ++i){
243         if(state)
244             this->readSector(sector_begin+i, out + i*sector_size);
245         else
246             this->readSector(sector_begin+i+_percentage.first, out + i*sector_size);
247     }
248 }
249

```

Có các đề xuất sau cho hướng tiếp cận trong việc thao tác với thông tin:

- Dựa vào thông tin đầu vào, tiến hành truy suất đến cluster tương ứng và chỉ lấy nội dung của cluster đó ra và chỉnh sửa. Yêu cầu thông tin đầu vào được qua các quá trình mã hóa và kiểm tra kỹ lưỡng. VD: chỉnh sửa thông tin của sv X thì phải biết trước đó sv X thì phải biết lần gần nhất sv X đã được lưu ở byte thứ bao nhiêu trong file 1. (quá trình thực hiện ko tiếp cận theo hướng này).
- Đọc hết toàn bộ thông tin tương ứng cần chỉnh sửa: truy xuất đến vị trí cần chỉnh sửa, tiến hành sửa và lưu lại. (chương trình demo được thực hiện theo hướng tiếp cận này).
- Việc thực hiện xóa/sửa được thiết kế tương tự như việc thêm file như bên dưới.

```

418 bool fileCombine::_add(const std::string& file_name, const std::string& pwd, const std::string& data){
419     char relate_pwd[16];
420     if(!check_valid_name(file_name)) return false;
421
422     ComputeMD5(pwd, relate_pwd);
423     if(!get_access_file1(relate_pwd) && !get_access_file2(relate_pwd)) return false;
424
425     if(!read_file(file_name, relate_pwd)) return false;
426
427     std::ofstream file("temp.txt", std::ios::app);
428
429     if (file.is_open()) {
430         file << data << std::endl;
431
432         file.close();
433
434         // std::cout << "Data has been written to temp.txt" << std::endl;
435     } else {
436         std::cerr << "Unable to open file: temp.txt" << std::endl;
437     }
438
439     if(!save_file(file_name, relate_pwd)) return false;
440
441     // read_file(file_name, relate_pwd);
442     return true;
443 }

```

Hình 7: Thêm đối tượng vào file tương ứng

2. Liệt kê một đoạn trong danh sách

- Thực hiện đọc toàn bộ file với tên và pwd tương ứng của file ra file tạm. Tùy vào mục đích cá nhân mà có thể tiến hành lọc, truy xuất từng cá nhân tương ứng và in ra màn hình.

```

445 bool fileCombine::_ls(const std::string& file_name, const std::string& pwd){
446     char relate_pwd[16];
447     if(!check_valid_name(file_name)) return false;
448
449     ComputeMD5(pwd, relate_pwd);
450
451     if(!get_access_file1(relate_pwd) && !get_access_file2(relate_pwd)) return false;
452
453     if(!read_file(file_name, relate_pwd)) return false;
454     return true;
455 }

```

3. Đổi mật khẩu

- Do yêu cầu không rõ ràng nên chỉ demo việc đổi mật khẩu cho file 1-2 (bên trong file kết hợp). Hoàn toàn có thể đổi mật khẩu cho file kết hợp theo phương pháp tương tự. (chỉ cần trở đúng đến vị trí lưu trữ mật khẩu của file tương ứng) và lưu ý một số chỗ liên quan đến mật khẩu cũ.

```

457 bool fileCombine::change_pwd(const std::string& name, const std::string& _old, const std::string& _new){
458     char relate_pwd[16];
459     if(!check_valid_name(name)) return false;
460
461     ComputeMD5(_old, relate_pwd);
462     if(!get_access_file1(relate_pwd) && !get_access_file2(relate_pwd)) return false;
463
464     if(strcmp(&name[0], combine_info+sector_size*2) == 0){
465         return change_pwd_file1(_new);
466     }
467     if(strcmp(&name[0], combine_info+sector_size*5) == 0){
468         return change_pwd_file2(_new);
469     }
470
471     return false;
472 }

```

4. Đổi Cơ chế mã hóa – hiện tại không hỗ trợ (do bị hạn chế về mặt thời gian).

Có các đề xuất sau cho việc sử dụng cơ chế mã hóa:

- Mã hóa từng đối tượng trước khi đưa vào file tạm sau đó mới tiến hành lưu file tạm vào file kết hợp.
- Khi có được file tạm lưu trữ các thông tin người dùng rồi, tiến hành mã hóa toàn bộ file tạm rồi mới lưu vào file kết hợp.

5. Nhận xét

Về mặt thiết kế của hệ thống đáp ứng mọi yêu cầu phụ cũng như yêu cầu chính.

Về mặt thực thi đáp ứng được các nhu cầu cơ bản cần thiết (thay đổi nội dung như thêm đối tượng, thay đổi mật khẩu, có cơ chế kiểm tra an toàn với mật khẩu, thao tác được với từng vùng nhớ và có thể kiểm soát từng vị trí cụ thể thông qua các hàm của sector và cluster hoặc tham chiếu trực tiếp đến địa chỉ được định nghĩa, ...) với khả năng mở rộng và tính ứng dụng cao.